aws

使用者指南

Amazon CloudWatch Logs

Amazon CloudWatch Logs: 使用者指南

Table of Contents

什麼是 Amazon CloudWatch Logs？

您可以使用 Amazon CloudWatch Logs 從 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、Route 53 和其他來源監控 AWS CloudTrail、存放和存取您的日誌檔案。

CloudWatch Logs 可讓您將您使用的所有系統、應用程式 AWS 和服務中的日誌集中在單一、高度可擴展的服務中。然後，您可以輕鬆檢視日誌、在日誌中搜尋特定的錯誤碼或模式、根據特定欄位篩選日誌，或安全封存日誌以供日後分析。CloudWatch Logs 可讓您將所有日誌 (無論其來源為何) 視為依時間排序的單一且一致的事件流量。

CloudWatch Logs 亦支援使用強大的查詢語言查詢日誌、稽核並遮罩日誌中的敏感資料，以及使用篩選條件或內嵌日誌格式從日誌產生指標。

CloudWatch Logs 支援兩個日誌類別。CloudWatch Logs 標準日誌類別中的日誌群組支援所有 CloudWatch Logs 功能。CloudWatch Logs 不常存取日誌類別中的日誌群組會產生較低的擷取費用，並支援一部分的標準類別功能。如需詳細資訊，請參閱日誌類別。

功能

- 兩種日誌類別具有彈性 – CloudWatch Logs 提供兩種日誌類別，因此您可以為不常存取的日誌提供經濟實惠的選項。對於需要即時監控或其他功能的日誌，您也有完整的功能選項。如需詳細資訊，請參閱日誌類別。
- 查詢您的日誌資料 – 您可以使用 CloudWatch Logs Insights 以互動方式搜尋和分析日誌資料。您可以執行查詢，協助您更有效率且有效地回應操作問題。CloudWatch Logs Insights 內建專用查詢語言，提供一些簡單又強大的命令。我們會提供範例查詢、命令描述、查詢自動完成及日誌欄位探索，以協助您開始使用。包含多種服務 AWS 日誌類型的範例查詢。若要開始使用，請參閱使用 CloudWatch Logs Insights 分析日誌資料。
- 建立欄位索引讓查詢更有效率 – 您可以在日誌事件中建立欄位索引。當您接著在 CloudWatch Logs Insights 查詢中使用欄位索引時，查詢會嘗試略過處理已知不包含索引欄位的日誌事件。此查詢可減少查詢的掃描量，讓您更快傳回結果。若要開始使用，請參閱建立欄位索引以改善查詢效能並減少掃描磁碟區。
- 使用 Live Tail 偵測和偵錯 – 您可以使用 Live Tail 在擷取時檢視新日誌事件的串流清單，快速進行事件疑難排解。可以近乎即時地檢視、篩選和反白顯示擷取的日誌，協助您快速偵測並解決問題。可以根據指定的詞彙篩選日誌，並反白顯示包含指定詞彙的日誌，以協助您快速找到查詢內容。如需詳細資訊，請參閱使用 CloudWatch Logs Live Tail 進行故障診斷。

- 從 Amazon EC2 執行個體監控日誌 - 您可以使用 CloudWatch Logs 透過日誌資料來監控應用程式和系統。例如，CloudWatch Logs 可以追蹤應用程式日誌中發生的錯誤數量，並在錯誤率超過您指定的閾值時通知您。CloudWatch Logs 會使用您的日誌資料進行監控，所以不需要變更程式碼。例如，您可以監控應用程式日誌特定的文字術語 (例如 "NullReferenceException") 或計算文字術語的出現日誌資料中的特定位置 (例如，在 Apache 存取日誌中的 "404" 狀態代碼)。找到您要搜尋的詞彙時，CloudWatch Logs 便會將資料回報至您指定的CloudWatch 指標。日誌資料會在移轉和靜態時加密。若要開始使用，請參閱開始使用 CloudWatch Logs。
- 監控 AWS CloudTrail 記錄的事件 – 您可以在 CloudWatch 中建立警示，並接收 CloudTrail 擷取的特定 API 活動的通知，並使用通知來執行故障診斷。若要開始使用，請參閱《AWS CloudTrail 使用者指南》中的將 CloudTrail 事件傳送至 CloudWatch Logs。
- 稽核並遮罩敏感資料 – 如果您的日誌中含有敏感資料，則可以利用資料保護政策協助您保護資料。這些政策可讓您稽核並遮罩敏感資料。如果您啟用資料保護功能，則系統會按預設遮罩與您選用之資料識別符相符的敏感資料。如需詳細資訊，請參閱使用遮罩功能協助保護敏感日誌資料。
- 日誌保留 - 根據預設，日誌將無限期保留且永遠不會過期。您可以調整每個日誌群組的保留政策，維持無限期保留，或選擇保留期間為 1 天至 10 年。
- 刪除保護 – 防止意外刪除日誌群組及其日誌串流的保護。在日誌群組上啟用時，刪除保護會封鎖所有刪除操作，直到明確停用為止。根據預設，不會啟用刪除保護。此選用功能有助於保護關鍵操作和合規資料免於意外移除，例如包含稽核資料的日誌群組，以及用於故障診斷和分析的生產應用程式日誌。
- 封存記錄資料 - 您可以使用 CloudWatch Logs 將日誌資料存放在高耐用性儲存中。CloudWatch Logs 代理程式可讓您輕鬆快速地將輪換和非輪換的日誌資料從主機傳送到日誌服務。然後，您可以在需要時存取原始日誌資料。
- 記錄 Route 53 DNS 查詢 - 您可以使用 CloudWatch Logs 來記錄 Route 53 所收到的 DNS 查詢的相關資訊。如需詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的記錄 DNS 查詢。
- 集中帳戶和區域的日誌 – 您可以使用 CloudWatch Logs Centralization 定義跨帳戶和跨區域集中規則，將跨多個帳戶和區域環境擷取的日誌資料複寫至中央區域和帳戶。您可以在中央帳戶中設定備份區域以提高彈性、為中央帳戶中新建立的日誌群組定義加密設定，以及使用增強型篩選功能，針對特定來源帳戶和區域的日誌事件建立指標和訂閱篩選條件。

相關 AWS 服務

以下服務係與 CloudWatch Logs 一起使用：

- AWS CloudTrail 是一種 Web 服務，可讓您監控對帳戶的 CloudWatch Logs API 發出的呼叫，包括 AWS 管理主控台、 AWS Command Line Interface (AWS CLI) 和其他 服務的呼叫。當 CloudTrail

記錄啟用時，CloudTrail 將擷取您帳戶中的 API 呼叫，並將日誌檔案傳送到您指定的 Amazon S3 儲存貯體。每個日誌檔案可以包含一個或多個記錄，取決於必須執行多少動作來滿足請求。如需詳細資訊 AWS CloudTrail，請參閱AWS CloudTrail 《 使用者指南》中的什麼是 AWS CloudTrail？。如需 CloudWatch 寫入 CloudTrail 日誌檔案的資料類型範例，請參閱在 中記錄 CloudWatch Logs API 和主控台操作 AWS CloudTrail。

- AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制使用者對 AWS 資源的存取。使用 IAM 控制誰可以使用您的 AWS 資源 (身分驗證)，以及他們可以透過何種方式使用哪些資源 (授權)。如需詳細資訊，請參閱《IAM 使用者指南》中的什麼是 IAM？。
- Amazon Kinesis Data Streams 這個 Web 服務可讓您快速且持續擷取和彙總資料。使用的資料類型包括 IT 基礎架構日誌資料、應用程式日誌、社交媒體、市場資料摘要和 web 點擊流資料。因為資料擷取和處理的回應時間是即時的，所以處理通常是輕量的。如需詳細資訊，請參閱《Amazon Kinesis Data Streams 開發人員指南》中的什麼是 Amazon Kinesis Data Streams？。
- AWS Lambda 是可讓您建置應用程式來快速回應新訊息的 web 服務。上傳您的應用程式碼當作 Lambda 函數，Lambda 會在高可用性運算基礎設施上執行您的程式碼，並完全負責管理運算資源，包括伺服器和作業系統維護、容量佈建與自動擴展、程式碼與安全性修補程式部署，以及程式碼監控和記錄。您唯一需要做的就是以 Lambda 支援的其中一種語言提供您的程式碼。如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的什麼是 AWS Lambda？。

定價

註冊時 AWS，您可以使用 AWS 免費方案免費開始使用 CloudWatch Logs。

其他使用 CloudWatch Logs 服務儲存的日誌 (例如，Amazon VPC 流程日誌和 Lambda 日誌) 採用標準費率。

如需定價的詳細資訊，請參閱 Amazon CloudWatch 定價。

如需有關如何分析 CloudWatch Logs 和 CloudWatch 的成本和用量，以及降低成本的最佳實務的詳細資訊，請參閱 CloudWatch 帳單與成本。

Amazon CloudWatch Logs 概念

下方描述您理解和使用 CloudWatch Logs 時的重要術語和概念。

日誌類別

CloudWatch Logs 提供兩種類別的日誌群組。標準日誌類別是日誌的完整功能選項，需要即時監控或您經常存取的日誌。不常存取日誌類別是您存取頻率較低的日誌的較低成本選項。它支援標準日誌類別功能的子集。

日誌事件

日誌事件是由應用程式或正被監控的資源來記錄的一些活動。CloudWatch Logs 理解的日誌事件記錄包含兩個屬性：事件發生的時間戳記及原始事件訊息。事件訊息必須為 UTF-8 編碼。

日誌串流

日誌串流是共享相同來源的一系列日誌事件。更具體地說，日誌串流通常旨在表示來自正在監視的應用程式執行個體或被監控的資源。例如，日誌串流可能與特定主機上的 Apache 存取日誌相關聯。當您不再需要日誌串流，您可以使用 aws logs delete-log-stream 命令將其刪除。

日誌群組

日誌群組定義了共享相同保留、監控和存取控制設定的日誌串流群組。每個日誌串流必須屬於一個日誌群組。例如，如果您為每個主機的 Apache 存取日誌提供單獨的日誌串流，您可以將這些日誌串流分到一個名為 `MyWebsite.com/Apache/access_log` 的日誌群組。

可以屬於一個日誌群組的日誌串流數量並沒有限制。

指標篩選條件

您可以使用指標篩選條件，從擷取的事件中擷取指標觀察值，並轉換為 CloudWatch 指標中的資料點。指標篩選條件指派給日誌群組，並且指派給日誌群組的所有篩選條件都會套用於其日誌串流。

保留設定

保留設定可用來指定日誌事件保留在 CloudWatch Logs 中多久。過期的日誌事件會自動刪除。就像指標篩選條件一樣，保留設定也會指派給日誌群組，並將指派給日誌群組的保留套用於其日誌串流。

刪除保護

刪除保護是一種保護機制，可防止意外刪除日誌群組及其日誌串流。在日誌群組上啟用時，刪除保護會封鎖所有刪除操作，直到明確停用為止。根據預設，不會啟用刪除保護。此選用功能有助於保護關鍵操作和合規資料免於意外移除，例如包含稽核資料的日誌群組，以及用於故障診斷和分析的生產應用程式日誌。

Amazon CloudWatch Logs 帳單與成本

如需有關如何分析 CloudWatch Logs 和 CloudWatch 的成本和用量，以及降低成本的最佳實務的詳細資訊，請參閱 CloudWatch 帳單與成本。

如需定價的詳細資訊，請參閱 Amazon CloudWatch 定價。

註冊時 AWS，您可以使用 AWS 免費方案免費開始使用 CloudWatch Logs。

其他使用 CloudWatch Logs 服務儲存的日誌 (例如，Amazon VPC 流程日誌和 Lambda 日誌) 採用標準費率。

日誌類別

CloudWatch Logs 提供兩種類別的日誌群組：

- CloudWatch Logs 標準日誌類別是日誌的完整功能選項，需要即時監控或您經常存取的日誌。
- CloudWatch Logs 不常存取日誌類別是新的日誌類別，可讓您以經濟實惠的方式合併日誌。此日誌類別提供 CloudWatch Logs 功能的子集，包括受管擷取、儲存、跨帳戶日誌分析，以及每 GB 較低擷取價格的加密。不常存取日誌類別非常適合對不常存取的日誌進行臨機操作查詢和after-the-fact鑑識分析。

> **Note**
>
> 對於費用，標準和不常存取日誌類別僅在擷取成本方面有所不同。儲存費用和 CloudWatch Logs Insights 費用在每個日誌類別中都相同。

如需 CloudWatch Logs 定價的詳細資訊，請參閱 Amazon CloudWatch 定價。

> **Important**
>
> 建立日誌群組之後，就無法變更其日誌類別。

支援的功能

下表列出每個日誌類別的功能。

功能	標準	不常存取	
全受管日誌擷取和儲存	是 ✓	是 ✓	
跨帳戶功能	是 ✓	是 ✓	
使用 加密 AWS KMS	是 ✓	是 ✓	
CloudWatch Logs Insights 查詢命令	是 ✓	是 ✓（大多數命令 – 請	

功能	標準	不常存取	
		參閱 日誌類別中支援的 Logs Insights QL 命令。)	
CloudWatch Logs Insights 探索的欄位	是 ✓	是 ✓	
面向	是 ✓	否	
使用 CloudWatch Pipeline 轉換日誌	是 ✓	是 ✓	
匯出至 Amazon S3	是 ✓	是 ✓	
S3 資料表整合	是 ✓	是 ✓	
排程查詢	是 ✓	是 ✓	
使用 OpenSearch PPL 或 OpenSearch SQL 在 CloudWatch Logs Insights 中查詢；	是 ✓	是 ✓	
自然語言查詢協助	是 ✓	否	
CloudWatch Logs 異常偵測	是 ✓	否	
Live Tail	是 ✓	否	
欄位索引	是 ✓	否	
與上一個時間範圍比較	是 ✓	否	
訂閱篩選條件	是 ✓	否	

功能	標準	不常存取
GetLogEvents 和 FilterLogEvents API 操作	是 ✓	不支援。使用 CloudWatch Logs Insights 檢視儲存在不常存取日誌類別中日誌群組中的日誌事件。
指標篩選條件	是 ✓	否
Container Insights 日誌擷取	是 ✓	否
Lambda Insights 日誌擷取	是 ✓	否
具有遮罩的敏感資料保護	是 ✓	是 ✓
內嵌指標格式	是 ✓	否

> **Note**
>
> 除了這兩個日誌類別之外，還有一個`Delivery`日誌類別。僅將`Delivery`日誌類別用於交付 AWS Lambda 日誌以存放在 Amazon S3 或 中 Amazon Data Firehose。交付類別中日誌群組中的日誌事件會保留在 CloudWatch Logs 中兩天。此保留期間是固定的，無法變更。此日誌類別不提供豐富的 CloudWatch Logs 功能，例如 CloudWatch Logs Insights 查詢。

開始使用 CloudWatch Logs

若要將來自 Amazon EC2 執行個體和內部部署伺服器的日誌收集到 CloudWatch Logs，請使用統一的 CloudWatch 代理程式。它可以讓您使用一個代理程式收集日誌及進階指標。它提供跨作業系統的支援，包括執行 Windows Server 的伺服器。此代理程式也提供更佳的效能。

如果您使用統一的 CloudWatch 代理程式來收集 CloudWatch 指標，它會啟用其他系統指標的收集，以提供訪客可見性。它也支援使用 `StatsD` 或 `collectd` 收集自訂指標。

如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的安裝 CloudWatch 代理程式。

舊版 CloudWatch Logs 代理程式 (僅支援從執行 Linux 的伺服器收集日誌) 已作廢且不再予以支援。有關從舊版 CloudWatch Logs 代理程式遷移到統一代理程式的資訊，請參閱藉助精靈建立 CloudWatch 代理程式組態檔案。

目錄

先決條件

若要使用 Amazon CloudWatch Logs，您需要 AWS 帳戶。 AWS 您的帳戶可讓您使用 服務 （例如 Amazon EC2) 來產生日誌，您可以在 Web 型界面 CloudWatch 主控台中檢視這些日誌。此外，您可以安裝和設定 AWS Command Line Interface (AWS CLI)。

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電或簡訊，並在電話鍵盤輸入驗證碼。

當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 https://aws.amazon.com/ 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立具有管理存取權的使用者

註冊 後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center和建立 管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS 管理主控台身分登入 。在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。
2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需說明，請參閱《IAM 使用者指南》中的為您的 AWS 帳戶 根使用者 （ 主控台 ） 啟用虛擬 MFA 裝置。

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的啟用 AWS IAM Identity Center。
2. 在 IAM Identity Center 中，將管理存取權授予使用者。

   如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center《 使用者指南》中的使用預設值設定使用者存取 IAM Identity Center 目錄。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《 使用者指南》中的登入 AWS 存取入口網站。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。
2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

設定命令列介面

您可以使用 AWS CLI 來執行 CloudWatch Logs 操作。

如需如何安裝和設定 的資訊 AWS CLI，請參閱AWS Command Line Interface 《 使用者指南》中的使用 AWS 命令列界面進行設定。

使用統一的 CloudWatch 代理程式來開始使用 CloudWatch Logs

如需有關使用統一的 CloudWatch 代理程式來開始使用 CloudWatch Logs 的相關資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。您完成此區段中所列的步驟以安裝、設定和啟動代理程式。如果您沒有將代理程式也用來收集 CloudWatch 指標，則可以忽略任何提到指標的章節。

如果您目前使用舊版 CloudWatch Logs 代理程式，且想要遷移為使用新的統一代理程式，我們建議您使用新代理程式套件中包含的精靈。這個精靈可以讀取目前的 CloudWatch Logs 代理程式組態檔案，並設定 CloudWatch 代理程式來收集相同的日誌。如需此精靈的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用精靈建立 CloudWatch 代理程式組態檔案。

使用之前的 CloudWatch 代理程式來開始使用 CloudWatch Logs

> **Important**
>
> CloudWatch 包含統一的 CloudWatch 代理程式，可從 EC2 執行個體和內部部署伺服器同時收集日誌和指標。舊版僅限日誌的代理程式已作廢且不再予以支援。

有關從舊版僅限日誌代理程式遷移到統一代理程式的資訊，請參閱藉助精靈建立 CloudWatch 代理程式組態檔案。
針對仍在使用舊版 CloudWatch Logs 代理程式的客戶，本節的其他部分說明該代理程式的使用方式。

您可以使用 CloudWatch Logs 代理程式，從執行 Linux 或 Windows Server 的 Amazon EC2 執行個體發佈日誌資料，以及發佈來自 AWS CloudTrail的記錄事件。我們建議改用 CloudWatch 統一代理程式來發佈日誌資料。如需新代理程式的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。

目錄

CloudWatch Logs 代理程式必要條件

CloudWatch Logs 代理程式需要 Python 2.7、3.0 或 3.3 版，以及下列任何版本的 Linux：

- Amazon Linux 2014.03.02 版或更新版本。不支援 Amazon Linux 2
- Ubuntu Server 12.04、14.04 或 16.04 版
- CentOS 版本 6、6.3、6.4、6.5 或 7.0
- Red Hat Enterprise Linux (RHEL) 版本 6.5 或 7.0
- Debian 8.0

快速入門：在執行中的 EC2 Linux 執行個體上安裝和設定 CloudWatch Logs 代理程式

Important

舊版 Logs 代理程式會被取代。CloudWatch 包含統一的代理程式，可從 EC2 執行個體和內部部署伺服器同時收集日誌和指標。如需詳細資訊，請參閱開始使用 CloudWatch Logs。
有關從舊版 CloudWatch Logs 代理程式遷移到統一代理程式的資訊，請參閱藉助精靈建立 CloudWatch 代理程式組態檔案。
舊版 Logs 代理程式只支援 2.6 到 3.5 版的 Python。此外，舊版 CloudWatch Logs 代理程式不支援執行個體中繼資料服務第 2 版 (IMDSv2)。如果伺服器使用 IMDSv2，您必須使用新版的統一代理程式，而不是舊版的 CloudWatch Logs 代理程式。
針對仍在使用舊版 CloudWatch Logs 代理程式的客戶，本節的其他部分說明該代理程式的使用方式。

Tip

CloudWatch 包含新的統一代理程式，可從 EC2 執行個體和內部部署伺服器同時收集日誌和指標。如果您尚未使用舊版 CloudWatch Logs 代理程式，我們建議您使用新版的統一 CloudWatch 代理程式。如需詳細資訊，請參閱開始使用 CloudWatch Logs。
此外，舊版的代理程式不支援執行個體中繼資料服務第 2 版 (IMDSv2)。如果伺服器使用 IMDSv2，您必須使用新版的統一代理程式，而不是舊版的 CloudWatch Logs 代理程式。
本節的其他部分說明舊版 CloudWatch Logs 代理程式的使用方式。

在執行中的 EC2 Linux 執行個體上設定舊版 CloudWatch Logs 代理程式

在現有的 EC2 執行個體上，您可以使用 CloudWatch Logs 代理程式安裝程式來安裝和設定 CloudWatch Logs 代理程式。安裝完成後，日誌會自動從執行個體提供到您在安裝代理程式時所建立的日誌串流。代理程式可確認它已啟動並保持執行，直到您停用為止。

除了使用代理程式之外，您也可以使用 AWS CLI、CloudWatch Logs SDK 或 CloudWatch Logs API 來發佈日誌資料。 AWS CLI 最適合在命令列或透過指令碼發佈資料。CloudWatch Logs 開發套件最適合直接從應用程式發佈日誌資料，或建置您自己的日誌發佈應用程式。

步驟 1：設定適用於 CloudWatch Logs 的 IAM 角色或使用者

CloudWatch Logs 代理程式支援 IAM 角色和使用者。如果您的執行個體已有相關聯的 IAM 角色，請務必包含以下的 IAM 政策。如果您尚無指派給執行個體的 IAM 角色，則可以在後續步驟中使用 IAM 憑證，或將 IAM 角色指派給該執行個體。如需詳細資訊，請參閱將 IAM 角色連接到執行個體。

設定適用於 CloudWatch Logs 的 IAM 角色或使用者

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 選取該角色名稱 (不選取該名稱旁的核取方塊) 以選擇角色。
4. 選擇 Attach Policies (連接政策)、Create Policy (建立政策)。

   新的瀏覽器標籤或視窗隨即開啟。
5. 選擇 JSON 標籤，並輸入下列 JSON 政策文件。

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
    ],
      "Resource": [
        "*"
    ]
  }
 ]
}
```

6. 完成時，選擇 Review policy (檢閱政策)。Policy Validator (政策檢查工具) 會回報任何語法錯誤。
7. 在 Review Policy (檢閱政策) 頁面上，為您正在建立的政策輸入 Name (名稱) 與 Description (描述) (選用)。檢閱政策 Summary (摘要) 來查看您的政策所授予的許可。然後選擇 Create policy (建立政策) 來儲存您的工作。

8. 關閉瀏覽器標籤或視窗，並返回您角色的 Add permissions (新增許可) 頁面。選擇 Refresh (重新整理)，然後選擇新政策以連接到您的角色。
9. 選擇 Attach Policy (連接政策)。

步驟 2：在現有的 Amazon EC2 執行個體上安裝和設定 CloudWatch Logs

根據 Amazon EC2 執行個體是執行 Amazon Linux、Ubuntu、CentOS 或 Red Hat 而定，安裝 CloudWatch Logs 代理程式的程序會有所不同。使用適用於您執行個體上 Linux 版本的步驟。

在現有的 Amazon Linux 執行個體上安裝和設定 CloudWatch Logs

從 Amazon Linux AMI 2014.09 開始，CloudWatch Logs 代理程式支援以 RPM 安裝 awslogs 套件。舊版的 Amazon Linux 可以使用 `sudo yum update -y` 命令更新其執行個體，以存取 awslogs 套件。透過使用 CloudWatch Logs 安裝程式將 awslogs 套件安裝為 RPM 而非 ，您的執行個體會從 接收定期套件更新和修補程式， AWS 而無需手動重新安裝 CloudWatch Logs 代理程式。

> **Warning**
>
> 如果您之前已使用 Python 指令碼來安裝 CloudWatch Logs 代理程式，請勿使用 RPM 安裝方法來更新代理程式。否則可能會導致組態問題，使得 CloudWatch Logs 代理程式無法將日誌傳送到 CloudWatch。

1. 連線至 Amazon Linux 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至您的執行個體。

   如需連線問題的詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至執行個體的故障診斷。
2. 更新 Amazon Linux 執行個體，以取得套件儲存庫中的最新變更。

   ```
   sudo yum update -y
   ```
3. 安裝 `awslogs` 套裝服務。這是在 Amazon Linux 執行個體上安裝 awslogs 的建議方法。

   ```
   sudo yum install -y awslogs
   ```
4. 編輯 `/etc/awslogs/awslogs.conf` 檔案來設定要追蹤的日誌。如需編輯此檔案的詳細資訊，請參閱 CloudWatch Logs 代理程式參考。

5. 根據預設，/etc/awslogs/awscli.conf 會指向 us-east-1 區域。若要將您的日誌推送至不同的區域，請編輯 awscli.conf 檔案並指定該區域。
6. 啟動 awslogs 服務。

   ```
   sudo service awslogs start
   ```

   如果您執行的是 Amazon Linux 2，請使用下列命令來啟動 awslogs 服務。

   ```
   sudo systemctl start awslogsd
   ```
7. (選用) 針對啟動服務時記錄的錯誤，核取 /var/log/awslogs.log 檔案。
8. (選用) 執行以下命令以在每次系統啟動時啟動 awslogs 服務。

   ```
   sudo chkconfig awslogs on
   ```

   如果您執行的是 Amazon Linux 2，請使用在每個系統啟動時的下列命令來啟動服務。

   ```
   sudo systemctl enable awslogsd.service
   ```
9. 在代理程式執行幾分鐘後，您在 CloudWatch 主控台應該會看到新建立的日誌群組和日誌串流。

   如需詳細資訊，請參閱檢視傳送到 CloudWatch Logs 的日誌資料。

在現有的 Ubuntu Server、CentOS 或 Red Hat 執行個體上安裝和設定 CloudWatch Logs

如果您使用的是執行 Ubuntu Server、CentOS、或 Red Hat 的 AMI，請使用下列程序，以手動在您的執行個體上安裝 CloudWatch Logs 代理程式。

1. 連線至 EC2 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至您的執行個體。

   如需連線問題的詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至執行個體的故障診斷。
2. 使用以下兩種選項其中之一來執行 CloudWatch Logs 代理程式安裝程式。您可以從網際網路直接執行或下載檔案並獨立執行。

> **Note**
>
> 如果您執行的是 CentOS 6.x、Red Hat 6.x 或 Ubuntu 12.04，請使用下載並執行獨立安裝程式的步驟。這些系統不支援直接從網際網路安裝 CloudWatch Logs 代理程式。

> **Note**
>
> 在 Ubuntu 上執行 `apt-get update`，再執行以下命令。

若要從網際網路直接執行，請使用下列命令並依照提示：

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
setup.py -O
```

```
sudo python ./awslogs-agent-setup.py --region us-east-1
```

如果上述命令不適用，請嘗試：

```
sudo python3 ./awslogs-agent-setup.py --region us-east-1
```

若要下載並單獨執行，請使用下列命令並依照提示：

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
setup.py -O
```

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/
AgentDependencies.tar.gz -O
```

```
tar xvf AgentDependencies.tar.gz -C /tmp/
```

```
sudo python ./awslogs-agent-setup.py --region us-east-1 --dependency-path /tmp/
AgentDependencies
```

您可以指定 us-east-1、us-west-1、us-west-2、ap-south-1、ap-northeast-2、ap-southeast-1、ap-southeast-2、ap-northeast-1、eu-central-1、eu-west-1 或 sa-east-1 區域來安裝 CloudWatch Logs 代理程式。

> **Note**
>
> 如需 `awslogs-agent-setup` 目前版本與版本歷史記錄的更多資訊，請參閱 CHANGELOG.txt。

CloudWatch Logs 代理程式安裝程式在設定期間需要特定的資訊。開始之前，您需要知道要監控的日誌檔及其時間戳記格式。您也應備妥下列資訊。

項目	Description
AWS 存取金鑰 ID	如果使用 IAM 角色，請按 Enter 鍵。否則，請輸入您的 AWS 存取金鑰 ID。
AWS 私密存取金鑰	如果使用 IAM 角色，請按 Enter 鍵。否則，請輸入您的 AWS 私密存取金鑰。
預設區域名稱	按 Enter。預設為 us-east-2。這可以設為 us-east-1、us-west-1、us-west-2、ap-south-1、ap-northeast-2、ap-southeast-1、ap-southeast-2、ap-northeast-1、eu-central-1、eu-west-1 或 sa-east-1。
預設輸出格式	保留空白並按 Enter 鍵。
要上傳的日誌檔路徑	包含要傳送之日誌資料的檔案位置。安裝程式會為您建議路徑。
Destination Log Group 名稱	您的日誌群組名稱。安裝程式會為您建議日誌群組名稱。
Destination Log Stream 名稱	在預設情況下，此為主機名稱。安裝程式會為您建議主機名稱。
時間戳記格式	指定在指定日誌檔中的時間戳記格式。選擇自訂以指定您自己的格式。

項目	Description
起始地位	如何上傳資料。將此設為 start_of_file 以上傳資料檔案中的所有項目。設定為 end_of_file 以只上傳新附加的資料。

在您完成這些步驟後，安裝程式會要求設定另一個日誌檔。您可以對每個日誌檔以您想要的次數多次執行程序。如果您已經沒有任何要監控的日誌檔，請在安裝程式提示您設定另一個日誌時選擇 N (否)。如需代理程式組態檔案中設定的詳細資訊，請參閱 CloudWatch Logs 代理程式參考。

> **Note**
>
> 不支援設定多個日誌來源，將資料傳送到單個日誌串流。

3. 在代理程式執行幾分鐘後，您在 CloudWatch 主控台應該會看到新建立的日誌群組和日誌串流。

   如需詳細資訊，請參閱檢視傳送到 CloudWatch Logs 的日誌資料。

快速入門：在 EC2 Linux 執行個體啟動時安裝和設定 CloudWatch Logs 代理程式

> **Tip**
>
> 本節討論的舊版 CloudWatch Logs 代理程式即將廢除。強烈建議您改用新的統一 CloudWatch 代理程式，以同時收集日誌和指標。此外，舊版 CloudWatch Logs 代理程式需要 Python 3.3 或更舊版本，而且根據預設，這些版本不會安裝在新的 EC2 執行個體上。如需統一 CloudWatch 代理程式的詳細資訊，請參閱安裝 CloudWatch 代理程式。
> 本節的其他部分說明舊版 CloudWatch Logs 代理程式的使用方式。

在 EC2 Linux 執行個體啟動時安裝舊版 CloudWatch Logs 代理程式

您可以使用 Amazon EC2 使用者資料 (這是 Amazon EC2 的一項功能，可在執行個體啟動時將參數資訊傳遞給執行個體)，在該執行個體上的安裝和設定 CloudWatch Logs 代理程式。若要將 CloudWatch Logs 代理程式的安裝和組態資訊傳遞給 Amazon EC2，您可以提供網路位置中的組態檔案 (例如 Amazon S3 儲存貯體)。

不支援設定多個日誌來源，將資料傳送到單個日誌串流。

先決條件

建立代理程式設定檔，其會描述您的所有日誌群組和日誌串流。這是一個文字檔案，其會描述要監控的日誌檔以及日誌群組和將要它們上傳至其中的日誌串流。代理程式會消耗此組態檔案，並開始監控和上傳所述的所有日誌檔。如需代理程式組態檔案中設定的詳細資訊，請參閱 CloudWatch Logs 代理程式參考。

以下是適用於 Amazon Linux 2 的代理程式組態檔案範例

```
[general]
state_file = /var/lib/awslogs/state/agent-state

[/var/log/messages]
file = /var/log/messages
log_group_name = /var/log/messages
log_stream_name = {instance_id}
datetime_format = %b %d %H:%M:%S
```

以下是適用於 Ubuntu 的範例代理程式組態檔案。

```
[general]
state_file = /var/awslogs/state/agent-state

[/var/log/syslog]
file = /var/log/syslog
log_group_name = /var/log/syslog
log_stream_name = {instance_id}
datetime_format = %b %d %H:%M:%S
```

設定 IAM 角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中選擇 Policies (政策)、Create Policy (建立政策)。
3. 在 Create Policy (建立政策) 頁面上，針對 Create Your Own Policy (建立您自己的政策)，選擇 Select (選取)。如需建立自訂政策的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 的 IAM 政策。 Amazon EC2
4. 在 Review Policy (檢閱政策) 頁面上的 Policy Name (政策名稱) 中，輸入該政策名稱。
5. 在 Policy Document (政策文件) 中，貼上以下政策：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

6. 選擇建立政策。
7. 在導覽窗格中，選擇 Roles (角色)、Create New Role (建立新角色)。
8. 在 Set Role Name (設定角色名稱) 頁面上，輸入該角色的名稱，然後選擇 Next Step (下一步)。
9. 在 Select Role Type (選取角色類型) 頁面上，選擇 Amazon EC2 旁的 Select (選取)。
10. 在 Attach Policy (連接政策) 頁面上的表格標頭中，選擇 Policy Type (政策類型)、Customer Managed (客戶受管理)。
11. 選取您建立的 IAM 政策，然後選擇 Next Step (下一步)。
12. 選擇建立角色。

如需使用者和政策的詳細資訊，請參閱《IAM 使用者指南》中的 IAM 使用者和群組以及管理 IAM 政策。

啟動新的執行個體和啟用 CloudWatch Logs

1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 選擇啟動執行個體。

   如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的啟動執行個體。 Amazon EC2
3. 在 Step 1: Choose an Amazon Machine Image (AMI) (步驟 1：選擇 Amazon Machine Image (AMI)) 頁面，選擇要啟動的 Linux 執行個體類型，然後在 Step 2: Choose an Instance Type (步驟 2：選擇執行個體類型) 頁面，選擇 Next: Configure Instance Details (下一步：設定執行個體詳細資料)。

   請確定 cloud-init 包含在您的 Amazon Machine Image (AMI) 中。Ubuntu 和 RHEL 的 Amazon Linux AMIs 和 AMIs 已包含 cloud-init，但 中的 CentOS 和其他 AMIs AWS Marketplace 可能沒有。
4. 在 Step 3: Configure Instance Details (步驟 3：設定執行個體詳細資訊) 頁面上，針對 IAM role (IAM 角色)，選取您建立的 IAM 角色。
5. 在 Advanced Details (進階詳細資訊)，針對 User data (使用者資料) 將以下指令碼貼到方塊中。然後，透過將 -c 選項的值變更為代理程式組態檔案的位置來更新該指令碼：

   ```
   #!/bin/bash
   curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
   setup.py -O
   chmod +x ./awslogs-agent-setup.py
   ./awslogs-agent-setup.py -n -r us-east-1 -c s3://amzn-s3-demo-bucket/my-config-file
   ```
6. 對執行個體進行其他變更、檢閱啟動設定，然後選擇 Launch (啟動)。
7. 在代理程式執行幾分鐘後，您在 CloudWatch 主控台應該會看到新建立的日誌群組和日誌串流。

   如需詳細資訊，請參閱檢視傳送到 CloudWatch Logs 的日誌資料。

快速入門：讓執行 Windows Server 2016 的 Amazon EC2 執行個體使用 CloudWatch Logs 代理程式將日誌傳送到 CloudWatch Logs

> **Tip**
>
> CloudWatch 包含新的統一代理程式，可從 EC2 執行個體和內部部署伺服器同時收集日誌和指標。我們建議您使用新版的統一 CloudWatch 代理程式。如需詳細資訊，請參閱開始使用 CloudWatch Logs。
> 本節的其他部分說明舊版 CloudWatch Logs 代理程式的使用方式。

讓執行 Windows Server 2016 的 Amazon EC2 執行個體使用舊版 CloudWatch Logs 代理程式將日誌傳送到 CloudWatch Logs

有多種方法可讓執行 Windows Server 2016 的執行個體將日誌傳送到 CloudWatch Logs。本節中的步驟使用 Systems Manager Run Command。如需其他可能的方法的詳細資訊，請參閱將日誌、事件，以及效能計數器傳送到 Amazon CloudWatch。

步驟

- 下載範例組態檔案
- 設定 CloudWatch 的 JSON 檔案
- 建立 Systems Manager 的 IAM 角色
- 驗證 Systems Manager 先決條件
- 驗證網際網路存取
- 使用 Systems Manager Run Command 以啟用 CloudWatch Logs

下載範例組態檔案

將以下範例檔案下載到您的電腦：`AWS.EC2.Windows.CloudWatch.json`。

設定 CloudWatch 的 JSON 檔案

在組態檔案中指定您的選擇，以決定將哪些日誌傳送到 CloudWatch。建立此檔案並指定選擇的程序。需要 30 分鐘或更久的時間完成。一旦完成此工作，您就可以在所有的執行個體上重複使用該組態檔案。

步驟

步驟 1：啟用 CloudWatch Logs

在 JSON 檔案頂部，針對 `IsEnabled` 將「false」變更為「true」。

```
"IsEnabled": true,
```

步驟 2：進行 CloudWatch 的設定

指定憑證、區域、日誌群組名稱和日誌串流命名空間。這可讓執行個體將日誌資料傳送到 CloudWatch Logs。若要將相同的日誌資料傳送到不同的位置，您可以使用唯一的 ID (例如 "CloudWatchLogs2" 和 "CloudWatchLogs3") 新增額外的區段，並為每個 ID 新增不同區域。

進行設定將日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 `CloudWatchLogs` 區段。

   ```
   {
       "Id": "CloudWatchLogs",
       "FullName":
    "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
       "Parameters": {
           "AccessKey": "",
           "SecretKey": "",
           "Region": "us-east-1",
           "LogGroup": "Default-Log-Group",
           "LogStream": "{instance_id}"
       }
   },
   ```

2. 將 `AccessKey` 和 `SecretKey` 欄位保留空白。使用 IAM 角色設定憑證。
3. 針對 `Region`，輸入日誌資料要送往的區域 (例如，`us-east-2`)。
4. 針對 `LogGroup`，輸入您日誌群組的名稱。此名稱會在 CloudWatch 主控台的 Log Groups (日誌群組) 畫面上出現。

5. 針對 LogStream，輸入目的地日誌串流。此名稱會在 CloudWatch 主控台的 Log Groups > Streams (日誌群組 > 串流) 畫面上出現。

   若使用 {instance_id} (預設值)，日誌串流名稱則為此執行個體的執行個體 ID。

   如果指定的日誌串流名稱尚未存在，CloudWatch Logs 會自動為您建立。您可以使用常值字串、預先定義的變數 {instance_id}、{hostname} 和 {ip_address}，或是組合這些項目，來定義日誌串流名稱。

步驟 3：設定要傳送的資料

您可以將事件日誌資料、Windows 事件追蹤 (ETW) 資料和其他日誌資料傳送至 CloudWatch Logs。

將 Windows 應用程式事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 ApplicationEventLog 區段。

```
{
    "Id": "ApplicationEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Application",
        "Levels": "1"
    }
},
```

2. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將安全性日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 SecurityEventLog 區段。

```
{
    "Id": "SecurityEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Security",
        "Levels": "7"
    }
},
```

2. 針對 Levels，輸入 **7** 以上傳所有訊息。

將系統事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 SystemEventLog 區段。

```
{
    "Id": "SystemEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "System",
        "Levels": "7"
    }
},
```

2. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將其他類型的事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中新增區段。每個區段必須有唯一的 Id。

```
{
    "Id": "Id-name",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Log-name",
        "Levels": "7"
    }
},
```

2. 針對 `Id`，輸入名稱代表要上傳的日誌 (例如，**`WindowsBackup`**)。
3. 針對 `LogName`，輸入要上傳日誌的名稱。您可以搜尋日誌的名稱，如下所示。

   a. 開啟事件檢視器。

   b. 在導覽窗格中，選擇 Applications and Services Logs (應用程式與服務日誌)。

   c. 導覽至日誌，然後選擇 Actions (動作)、Properties (屬性)。
4. 針對 `Levels`，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將 Windows 事件追蹤資料傳送到 CloudWatch Logs

ETW (Windows 的事件追蹤功能) 提供有效率和詳細的記錄機制，應用程式可將日誌寫入其中。每個 ETW 皆透過工作階段管理員控制，該管理程式可啟動和停止記錄工作階段。每個工作階段都有提供者和一個或多個使用者。

1. 在 JSON 檔案中，找到 `ETW` 區段。

```
{
    "Id": "ETW",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
```

```
        "LogName": "Microsoft-Windows-WinINet/Analytic",
        "Levels": "7"
    }
},
```

2. 針對 LogName，輸入要上傳日誌的名稱。
3. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將自訂日誌 (任何文字型日誌檔案) 傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 CustomLogs 區段。

```
{
    "Id": "CustomLogs",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogDirectoryPath": "C:\\CustomLogs\\",
        "TimestampFormat": "MM/dd/yyyy HH:mm:ss",
        "Encoding": "UTF-8",
        "Filter": "",
        "CultureName": "en-US",
        "TimeZoneKind": "Local",
        "LineCount": "5"
    }
},
```

2. 針對 LogDirectoryPath，輸入在執行個體上存放日誌的路徑。
3. 針對 TimestampFormat，輸入要使用的時間戳記格式。如需關於支援之值的詳細資訊，請參閱 MSDN 上的自訂日期與時間格式字串主題。

> **Important**
>
> 您的原始日誌檔在每個日誌行的開始都必須有時間戳記，且在時間戳記後必須接著一個空格。

4. 針對 Encoding，輸入要使用的檔案編碼 (例如，UTF-8)。如需支援值的清單，請參閱 MSDN 上的 Encoding 類別主題。

> **Note**
>
> 使用編碼名稱，而非顯示名稱。

5. (選用) 針對 Filter，輸入日誌名稱的前綴。將此參數留白，以監控所有檔案。如需關於支援之值的詳細資訊，請參閱 MSDN 上的 FileSystemWatcherFilter 屬性主題。
6. (選用) 針對 CultureName，輸入要記錄時間戳記的地區設定。若 CultureName 留空，其會預設為與 Windows 執行個體目前使用的相同地區設定。如需詳細資訊，請參閱 MSDN 中Language tag產品行為主題中表格的 欄。

> **Note**
>
> 不支援 div、div-MV、hu 和 hu-HU 值。

7. (選用) 針對 TimeZoneKind，請輸入 Local 或 UTC。您可以設定此以在日誌時間戳記未包含任何時區資訊時提供時區資訊。如果此參數空白，且如果您的時間戳記不包含時區資訊，則 CloudWatch Logs 預設為本機時區。如果您的時間戳記已包含時區資訊，則會忽略此參數。
8. (選用) 針對 LineCount，輸入標頭中的行數，以辨識日誌檔案。例如，IIS 日誌檔幾乎都具有相同的標頭。您可以輸入 **5**，這會讀取日誌檔標頭的前三行來進行辨識。在 IIS 日誌檔中，第三行是日期和時間戳記，但不保證時間戳記在日誌檔間總是不同的。因此，我們建議包括至少一行實際的日誌資料，做為用來唯一辨識日誌檔的指紋。

將安全性日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 IISLog 區段。

```
{
    "Id": "IISLogs",
```

```
    "FullName":
 "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogDirectoryPath": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
        "TimestampFormat": "yyyy-MM-dd HH:mm:ss",
        "Encoding": "UTF-8",
        "Filter": "",
        "CultureName": "en-US",
        "TimeZoneKind": "UTC",
        "LineCount": "5"
    }
},
```

2. 針對 LogDirectoryPath，輸入針對個別網站儲存 IIS 日誌的資料夾 (例如，C:\inetpub\logs\LogFiles\W3SVC*n*)。

> **Note**
>
> 只支援 W3C 日誌格式。不支援 IIS、NCSA 和自訂格式。

3. 針對 TimestampFormat，輸入要使用的時間戳記格式。如需關於支援之值的詳細資訊，請參閱 MSDN 上的自訂日期與時間格式字串主題。
4. 針對 Encoding，輸入要使用的檔案編碼 (例如，UTF-8)。如需關於支援之值的詳細資訊，請參閱 MSDN 上的編碼類別主題。

> **Note**
>
> 使用編碼名稱，而非顯示名稱。

5. (選用) 針對 Filter，輸入日誌名稱的前綴。將此參數留白，以監控所有檔案。如需關於支援之值的詳細資訊，請參閱 MSDN 上的 FileSystemWatcherFilter 屬性主題。
6. (選用) 針對 CultureName，輸入要記錄時間戳記的地區設定。若 CultureName 留空，其會預設為與 Windows 執行個體目前使用的相同地區設定。如需支援之值的詳細資訊，請參閱 MSDN 中Language tag產品行為主題中表格的 欄。

> **Note**
>
> 不支援 div、div-MV、hu 和 hu-HU 值。

7. (選用) 針對 TimeZoneKind，輸入 Local 或 UTC。您可以設定此以在日誌時間戳記未包含任何時區資訊時提供時區資訊。如果此參數空白，且如果您的時間戳記不包含時區資訊，則 CloudWatch Logs 預設為本機時區。如果您的時間戳記已包含時區資訊，則會忽略此參數。
8. (選用) 針對 LineCount，輸入標頭中的行數，以辨識日誌檔案。例如，IIS 日誌檔幾乎都具有相同的標頭。您可以輸入 **5**，這會讀取日誌檔標頭的前五行來進行辨識。在 IIS 日誌檔中，第三行是日期和時間戳記，但不保證時間戳記在日誌檔間總是不同的。因此，我們建議包括至少一行實際的日誌資料，以唯一辨識日誌檔的指紋。

步驟 4：設定流程控制

每個資料類型必須擁有在 Flows 區段中的相對應目的地。例如，若要將自訂日誌、ETW 日誌和系統日誌傳送到 CloudWatch Logs，請將 (CustomLogs,ETW,SystemEventLog),CloudWatchLogs 新增至 Flows 區段。

> **Warning**
>
> 加入無效的步驟會阻礙流程。例如，如果新增磁碟指標步驟，但執行個體沒有磁碟，則流程中的所有步驟都會遭到封鎖。

您可以將相同的日誌檔傳送到多個目的地。例如，如果要將應用程式日誌傳送到您在 CloudWatchLogs 區段中所定義的兩個不同目的地，請在 Flows 區段中加入 ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)。

設定流程控制

1. 在 AWS.EC2.Windows.CloudWatch.json 檔案中，找到 Flows 區段。

```
"Flows": {
    "Flows": [
      "PerformanceCounter,CloudWatch",
      "(PerformanceCounter,PerformanceCounter2), CloudWatch2",
      "(CustomLogs, ETW, SystemEventLog),CloudWatchLogs",
      "CustomLogs, CloudWatchLogs2",
      "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"
    ]
}
```

2. 針對 Flows，新增要上傳的每個資料類型 (例如，ApplicationEventLog) 及其目的地 (例如，CloudWatchLogs)。

步驟 5：儲存 JSON 內容

您現在已經完成編輯 JSON 檔案。進行儲存，並將檔案內容貼到另一個視窗中的文字編輯器。您將需要此程序在後續步驟中的檔案內容。

建立 Systems Manager 的 IAM 角色

當您使用 Systems Manager Run Command 時，需要執行個體憑證的 IAM 角色。此角色可讓 Systems Manager 在執行個體上執行動作。如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的設定 Systems Manager 的安全性角色。如需如何將 IAM 角色連接至現有執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的將 IAM 角色連接至執行個體。

驗證 Systems Manager 先決條件

使用 Systems Manager Run Command 來設定與 CloudWatch Logs 整合之前，請確認您的執行個體符合最低要求。如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Systems Manager 先決條件。

驗證網際網路存取

Amazon EC2 Windows Server 執行個體和受管執行個體必須有對外網際網路存取，才能將日誌和事件資料傳送到 CloudWatch。如需有關如何設定網際網路存取的詳細資訊，請參閱《Amazon VPC 使用者指南》中的網際網路閘道。

使用 Systems Manager Run Command 以啟用 CloudWatch Logs

Run Command (執行指令) 可讓您隨需管理執行個體的組態。您指定 Systems Manager 文件、指定參數，然後在一或多個執行個體上執行命令。執行個體上的 SSM Agent 會依指定來處理命令和設定執行個體。

使用 Run Command 設定與 CloudWatch Logs 整合

1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 開啟位於 https://console.aws.amazon.com/systems-manager/ 的 SSM 主控台。
3. 在導覽窗格中，選擇 執行命令。
4. 選擇 Run a command (執行指令)。
5. 針對 Command document (指令文件)，選擇 AWS-ConfigureCloudWatch。

6. 針對 Target instances (目標執行個體)，選擇要與 CloudWatch Logs 整合的執行個體。如果執行個體未出現於此清單中，可能是無法針對 Run Command 進行設定。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Systems Manager 先決條件。
7. 針對 Status (狀態)，請選擇 Enabled (啟用)。
8. 對於 Properties (屬性)，複製並貼上您在之前任務建立的 JSON 內容。
9. 完成填寫剩下的選填欄位，然後選擇 Run (執行)。

使用下列程序，在 Amazon EC2 主控台檢視命令執行的結果。

在主控台中檢視指令輸出

1. 選取指令。
2. 選擇 Output (輸出) 索引標籤。
3. 選擇 View Output (檢視輸出)。此指令輸出頁面會顯示指令執行的結果。

快速入門：讓執行 Windows Server 2012 和 Windows Server 2008 的 Amazon EC2 執行個體將日誌傳送到 CloudWatch Logs

> **Tip**
>
> CloudWatch 包含新的統一代理程式，可從 EC2 執行個體和內部部署伺服器同時收集日誌和指標。我們建議您使用新版的統一 CloudWatch 代理程式。如需詳細資訊，請參閱開始使用 CloudWatch Logs。
> 本節的其他部分說明舊版 CloudWatch Logs 代理程式的使用方式。

讓執行 Windows Server 2012 和 Windows Server 2008 的 Amazon EC2 執行個體將日誌傳送到 CloudWatch Logs

使用以下步驟讓執行 Windows Server 2012 和 Windows Server 2008 的執行個體將日誌傳送到 CloudWatch Logs。

下載範例組態檔案

將以下範例 JSON 檔案下載到您的電腦：`AWS.EC2.Windows.CloudWatch.json`。在以下步驟進行編輯。

設定 CloudWatch 的 JSON 檔案

在 JSON 組態檔案中指定您的選擇，以決定將哪些日誌傳送到 CloudWatch。建立此檔案並指定選擇的程序。需要 30 分鐘或更久的時間完成。一旦完成此工作，您就可以在所有的執行個體上重複使用該組態檔案。

步驟

步驟 1：啟用 CloudWatch Logs

在 JSON 檔案頂部，針對 `IsEnabled` 將「false」變更為「true」。

```
"IsEnabled": true,
```

步驟 2：進行 CloudWatch 的設定

指定憑證、區域、日誌群組名稱和日誌串流命名空間。這可讓執行個體將日誌資料傳送到 CloudWatch Logs。若要將相同的日誌資料傳送到不同的位置，您可以使用唯一的 ID (例如 "CloudWatchLogs2" 和 "CloudWatchLogs3") 新增額外的區段，並為每個 ID 新增不同區域。

進行設定將日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 `CloudWatchLogs` 區段。

```
{
    "Id": "CloudWatchLogs",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "AccessKey": "",
        "SecretKey": "",
        "Region": "us-east-1",
        "LogGroup": "Default-Log-Group",
        "LogStream": "{instance_id}"
    }
},
```

2. 將 AccessKey 和 SecretKey 欄位保留空白。使用 IAM 角色設定憑證。
3. 針對 Region，輸入日誌資料要送往的區域 (例如，us-east-2)。
4. 針對 LogGroup，輸入您日誌群組的名稱。此名稱會在 CloudWatch 主控台的 Log Groups (日誌群組) 畫面上出現。
5. 針對 LogStream，輸入目的地日誌串流。此名稱會在 CloudWatch 主控台的 Log Groups > Streams (日誌群組 > 串流) 畫面上出現。

   若使用 {instance_id} (預設值)，日誌串流名稱則為此執行個體的執行個體 ID。

   如果指定的日誌串流名稱尚未存在，CloudWatch Logs 會自動為您建立。您可以使用常值字串、預先定義的變數 {instance_id}、{hostname} 和 {ip_address}，或是組合這些項目，來定義日誌串流名稱。

步驟 3：設定要傳送的資料

您可以將事件日誌資料、Windows 事件追蹤 (ETW) 資料和其他日誌資料傳送至 CloudWatch Logs。

將 Windows 應用程式事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 ApplicationEventLog 區段。

```
{
    "Id": "ApplicationEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Application",
        "Levels": "1"
    }
},
```

2. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將安全性日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 SecurityEventLog 區段。

```
{
    "Id": "SecurityEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Security",
        "Levels": "7"
    }
},
```

2. 針對 Levels，輸入 **7** 以上傳所有訊息。

將系統事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 SystemEventLog 區段。

```
{
    "Id": "SystemEventLog",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "System",
        "Levels": "7"
    }
},
```

2. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將其他類型的事件日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中新增區段。每個區段必須有唯一的 `Id`。

```
{
    "Id": "Id-name",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Log-name",
        "Levels": "7"
    }
},
```

2. 針對 `Id`，輸入名稱代表要上傳的日誌 (例如，**`WindowsBackup`**)。
3. 針對 `LogName`，輸入要上傳日誌的名稱。您可以搜尋日誌的名稱，如下所示。
   a. 開啟事件檢視器。
   b. 在導覽窗格中，選擇 Applications and Services Logs (應用程式與服務日誌)。
   c. 導覽至日誌，然後選擇 Actions (動作)、Properties (屬性)。
4. 針對 `Levels`，指定要上傳訊息的類型。您可以指定下列其中一個值：
   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將 Windows 事件追蹤資料傳送到 CloudWatch Logs

ETW (Windows 的事件追蹤功能) 提供有效率和詳細的記錄機制，應用程式可將日誌寫入其中。每個 ETW 皆透過工作階段管理員控制，該管理程式可啟動和停止記錄工作階段。每個工作階段都有提供者和一個或多個使用者。

1. 在 JSON 檔案中，找到 `ETW` 區段。

```
{
```

```
    "Id": "ETW",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogName": "Microsoft-Windows-WinINet/Analytic",
        "Levels": "7"
    }
},
```

2. 針對 LogName，輸入要上傳日誌的名稱。
3. 針對 Levels，指定要上傳訊息的類型。您可以指定下列其中一個值：

   - **1** - 僅上傳錯誤訊息。
   - **2** - 僅上傳警告訊息。
   - **4** - 僅上傳資訊訊息。

   您可以將值組合，以包含多個類型訊息。例如，值為 **3** 會上傳錯誤訊息 (**1**) 和警告訊息 (**2**)。值為 **7** 會上傳錯誤訊息 (**1**)、警告訊息 (**2**) 和資訊訊息 (**4**)。

將自訂日誌 (任何文字型日誌檔案) 傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 CustomLogs 區段。

```
{
    "Id": "CustomLogs",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogDirectoryPath": "C:\\CustomLogs\\",
        "TimestampFormat": "MM/dd/yyyy HH:mm:ss",
        "Encoding": "UTF-8",
        "Filter": "",
        "CultureName": "en-US",
        "TimeZoneKind": "Local",
        "LineCount": "5"
    }
},
```

2. 針對 LogDirectoryPath，輸入在執行個體上存放日誌的路徑。

3. 針對 `TimestampFormat`，輸入要使用的時間戳記格式。如需關於支援之值的詳細資訊，請參閱 MSDN 上的自訂日期與時間格式字串主題。

   > **Important**
   >
   > 您的原始日誌檔在每個日誌行的開始都必須有時間戳記，且在時間戳記後必須接著一個空格。

4. 針對 `Encoding`，輸入要使用的檔案編碼 (例如，UTF-8)。如需關於支援之值的詳細資訊，請參閱 MSDN 上的編碼類別主題。

   > **Note**
   >
   > 使用編碼名稱，而非顯示名稱。

5. (選用) 針對 `Filter`，輸入日誌名稱的前綴。將此參數留白，以監控所有檔案。如需關於支援之值的詳細資訊，請參閱 MSDN 上的 FileSystemWatcherFilter 屬性主題。
6. (選用) 針對 `CultureName`，輸入要記錄時間戳記的地區設定。若 `CultureName` 留空，其會預設為與 Windows 執行個體目前使用的相同地區設定。如需支援之值的詳細資訊，請參閱 MSDN 中`Language tag`產品行為主題中表格的 欄。

   > **Note**
   >
   > 不支援 `div`、`div-MV`、`hu` 和 `hu-HU` 值。

7. (選用) 針對 `TimeZoneKind`，請輸入 `Local` 或 `UTC`。您可以設定此以在日誌時間戳記未包含任何時區資訊時提供時區資訊。如果此參數空白，且如果您的時間戳記不包含時區資訊，則 CloudWatch Logs 預設為本機時區。如果您的時間戳記已包含時區資訊，則會忽略此參數。
8. (選用) 針對 `LineCount`，輸入標頭中的行數，以辨識日誌檔案。例如，IIS 日誌檔幾乎都具有相同的標頭。您可以輸入 **5**，這會讀取日誌檔標頭的前三行來進行辨識。在 IIS 日誌檔中，第三行是日期和時間戳記，但不保證時間戳記在日誌檔間總是不同的。因此，我們建議包括至少一行實際的日誌資料，做為用來唯一辨識日誌檔的指紋。

將安全性日誌資料傳送到 CloudWatch Logs

1. 在 JSON 檔案中，找到 `IISLog` 區段。

```
{
```

```
    "Id": "IISLogs",
    "FullName":
 "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
    "Parameters": {
        "LogDirectoryPath": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
        "TimestampFormat": "yyyy-MM-dd HH:mm:ss",
        "Encoding": "UTF-8",
        "Filter": "",
        "CultureName": "en-US",
        "TimeZoneKind": "UTC",
        "LineCount": "5"
    }
},
```

2. 針對 LogDirectoryPath，輸入針對個別網站儲存 IIS 日誌的資料夾 (例如，C:\inetpub\logs\LogFiles\W3SVC*n*)。

> **Note**
>
> 只支援 W3C 日誌格式。不支援 IIS、NCSA 和自訂格式。

3. 針對 TimestampFormat，輸入要使用的時間戳記格式。如需關於支援之值的詳細資訊，請參閱 MSDN 上的自訂日期與時間格式字串主題。
4. 針對 Encoding，輸入要使用的檔案編碼 (例如，UTF-8)。如需關於支援之值的詳細資訊，請參閱 MSDN 上的編碼類別主題。

> **Note**
>
> 使用編碼名稱，而非顯示名稱。

5. (選用) 針對 Filter，輸入日誌名稱的前綴。將此參數留白，以監控所有檔案。如需關於支援之值的詳細資訊，請參閱 MSDN 上的 FileSystemWatcherFilter 屬性主題。
6. (選用) 針對 CultureName，輸入要記錄時間戳記的地區設定。若 CultureName 留空，其會預設為與 Windows 執行個體目前使用的相同地區設定。如需支援之值的詳細資訊，請參閱 MSDN 中Language tag產品行為主題中表格的 欄。

> **Note**
>
> 不支援 div、div-MV、hu 和 hu-HU 值。

7. (選用) 針對 TimeZoneKind，輸入 Local 或 UTC。您可以設定此以在日誌時間戳記未包含任何時區資訊時提供時區資訊。如果此參數空白，且如果您的時間戳記不包含時區資訊，則 CloudWatch Logs 預設為本機時區。如果您的時間戳記已包含時區資訊，則會忽略此參數。
8. (選用) 針對 LineCount，輸入標頭中的行數，以辨識日誌檔案。例如，IIS 日誌檔幾乎都具有相同的標頭。您可以輸入 **5**，這會讀取日誌檔標頭的前五行來進行辨識。在 IIS 日誌檔中，第三行是日期和時間戳記，但不保證時間戳記在日誌檔間總是不同的。因此，我們建議包括至少一行實際的日誌資料，以唯一辨識日誌檔的指紋。

步驟 4：設定流程控制

每個資料類型必須擁有在 Flows 區段中的相對應目的地。例如，若要將自訂日誌、ETW 日誌和系統日誌傳送到 CloudWatch Logs，請將 (CustomLogs,ETW,SystemEventLog),CloudWatchLogs 新增至 Flows 區段。

> **Warning**
>
> 加入無效的步驟會阻礙流程。例如，如果新增磁碟指標步驟，但執行個體沒有磁碟，則流程中的所有步驟都會遭到封鎖。

您可以將相同的日誌檔傳送到多個目的地。例如，如果要將應用程式日誌傳送到您在 CloudWatchLogs 區段中所定義的兩個不同目的地，請在 Flows 區段中加入 ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)。

設定流程控制

1. 在 AWS.EC2.Windows.CloudWatch.json 檔案中，找到 Flows 區段。

```
"Flows": {
    "Flows": [
      "PerformanceCounter,CloudWatch",
      "(PerformanceCounter,PerformanceCounter2), CloudWatch2",
      "(CustomLogs, ETW, SystemEventLog),CloudWatchLogs",
      "CustomLogs, CloudWatchLogs2",
      "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"
    ]
}
```

2. 針對 Flows，新增要上傳的每個資料類型 (例如，ApplicationEventLog) 及其目的地 (例如，CloudWatchLogs)。

您現在已經完成編輯 JSON 檔案。在後續步驟中會使用到此檔案。

啟動代理程式

若要讓執行 Windows Server 2012 或 Windows Server 2008 的 Amazon EC2 執行個體將日誌傳送到 CloudWatch Logs，請使用 EC2Config 服務 (EC2Config.exe)。您的執行個體應該擁有 EC2Config 4.0 或更新版本，而且您可以使用此程序。

使用 EC2Config 4.x 來設定 CloudWatch

1. 對您稍早在此程序中編輯的 AWS.EC2.Windows.CloudWatch.json 檔案檢查編碼。只支援無 BOM 的 UTF-8 編碼。接著，在 Windows Server 2008 – 2012 R2 執行個體的以下資料夾儲存檔案：C:\Program Files\Amazon\SSM\Plugins\awsCloudWatch\。
2. 使用 Windows 服務控制台，或使用下列 PowerShell 命令，以啟動或重新啟動 SSM Agent (AmazonSSMAgent.exe)：

```
PS C:\> Restart-Service AmazonSSMAgent
```

SSM Agent 重新啟動之後會偵測組態檔案，並設定 CloudWatch 整合的執行個體。如果變更本機組態檔案中的參數與設定，您需要重新啟動 SSM Agent 來反映變更。若要停用執行個體上的 CloudWatch 整合，請將 IsEnabled 變更為 false，並儲存組態檔案中的變更。

報告 CloudWatch Logs 代理程式狀態

請使用下列程序來報告 CloudWatch Logs 代理程式在 EC2 執行個體上的狀態。

報告代理程式的狀態

1. 連線至 EC2 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至您的執行個體。

   如需連線問題的詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至執行個體的故障診斷
2. 在命令提示，請輸入下列命令：

```
sudo service awslogs status
```

如果您執行的是 Amazon Linux 2，請輸入下列命令：

```
sudo service awslogsd status
```

3. 檢查 /var/log/awslogs.log 檔案中是否有任何與 CloudWatch Logs 代理程式相關的錯誤、警告或問題。

啟動 CloudWatch Logs 代理程式

如果 EC2 執行個體上的 CloudWatch Logs 代理程式在安裝後沒有自動啟動，或您已停止代理程式，您可以使用以下程序來啟動代理程式。

啟動代理程式

1. 連線至 EC2 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至您的執行個體。

   如需連線問題的詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至執行個體的故障診斷。
2. 在命令提示，請輸入下列命令：

   ```
   sudo service awslogs start
   ```

   如果您執行的是 Amazon Linux 2，請輸入下列命令：

   ```
   sudo service awslogsd start
   ```

停止 CloudWatch Logs 代理程式

請使用下列程序來停止在 EC2 執行個體上的 CloudWatch Logs 代理程式。

停止代理程式

1. 連線至 EC2 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至您的執行個體。

如需連線問題的詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至執行個體的故障診斷。

2. 在命令提示，請輸入下列命令：

```
sudo service awslogs stop
```

如果您執行的是 Amazon Linux 2，請輸入下列命令：

```
sudo service awslogsd stop
```

CloudWatch Logs 代理程式參考

> **Important**
>
> 本節是使用已棄用 CloudWatch Logs 代理程式的參考。如果您使用的是執行個體中繼資料服務第 2 版 (IMDSv2)，則必須使用新的統一 CloudWatch 代理程式。不過，即使您不是使用 IMDSv2，我們強烈建議使用較新的統一 CloudWatch 代理程式，而不是已取代的 CloudWatch Logs 代理程式。如需有關較新的統一 CloudWatch 代理程式的資訊，請參閱使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。如需有關從已棄用 CloudWatch Logs 代理程式遷移至統一代理程式的資訊，請使用精靈建立 CloudWatch 代理程式組態檔案。

CloudWatch Logs 代理程式可自動將日誌資料從 Amazon EC2 執行個體傳送至 CloudWatch Logs。代理程式包含下列元件：

- 將日誌資料 AWS CLI 推送至 CloudWatch Logs 的 外掛程式。
- 指令碼 (常駐程式)，起始將資料推送至 CloudWatch Logs 的程序。
- 確保協助程式持續執行的 Cron 工作。

代理程式組態檔案

CloudWatch Logs 代理程式組態檔案描述 CloudWatch Logs 代理程式所需的資訊。代理程式組態檔案的 [general] 區段定義了通用組態，而這些組態會套用到所有日誌串流。[logstream] 區段定義了將本機檔案傳送到遠端日誌串流時所需的資訊。您可以擁有多個 [logstream] 區段，但每個區段在組態檔案中

都必須有唯一的名稱，例如，[logstream1]、[logstream2]，以此類推。[logstream] 值與日誌檔中的第一行資料，用於定義日誌檔的身分。

```
[general]
state_file = value
logging_config_file = value
use_gzip_http_content_encoding = [true | false]

[logstream1]
log_group_name = value
log_stream_name = value
datetime_format = value
time_zone = [LOCAL|UTC]
file = value
file_fingerprint_lines = integer | integer-integer
multi_line_start_pattern = regex | {datetime_format}
initial_position = [start_of_file | end_of_file]
encoding = [ascii|utf_8|..]
buffer_duration = integer
batch_count = integer
batch_size = integer

[logstream2]
...
```

state_file

指定狀態檔案的存放位置。

logging_config_file

(選用) 指定代理程式日誌組態檔案的位置。如果您未在此指定代理程式日誌組態檔案，將使用預設檔案 awslogs.conf。如果您以指令碼安裝代理程式，預設的檔案位置是 `/var/awslogs/etc/awslogs.conf`，如果以 rpm 安裝代理程式，則是 `/etc/awslogs/awslogs.conf`。此檔案為 Python 組態檔案格式 (https://docs.python.org/2/library/logging.config.html#logging-config-fileformat)。您可以自訂具有以下名稱的記錄器。

```
cwlogs.push
cwlogs.push.reader
cwlogs.push.publisher
cwlogs.push.event
cwlogs.push.batch
```

```
cwlogs.push.stream
cwlogs.push.watcher
```

以下範例會將讀取者和發佈者的層級變更為 WARNING，而預設值為 INFO。

```
[loggers]
keys=root,cwlogs,reader,publisher

[handlers]
keys=consoleHandler

[formatters]
keys=simpleFormatter

[logger_root]
level=INFO
handlers=consoleHandler

[logger_cwlogs]
level=INFO
handlers=consoleHandler
qualname=cwlogs.push
propagate=0

[logger_reader]
level=WARNING
handlers=consoleHandler
qualname=cwlogs.push.reader
propagate=0

[logger_publisher]
level=WARNING
handlers=consoleHandler
qualname=cwlogs.push.publisher
propagate=0

[handler_consoleHandler]
class=logging.StreamHandler
level=INFO
formatter=simpleFormatter
args=(sys.stderr,)

[formatter_simpleFormatter]
```

```
format=%(asctime)s - %(name)s - %(levelname)s - %(process)d - %(threadName)s -
 %(message)s
```

use_gzip_http_content_encoding

設為 true (預設) 時會啟用 gzip http 內容編碼，將壓縮的酬載傳送至 CloudWatch Logs。這會降低 CPU 使用量、降低 NetworkOut 及減少推送延遲。若要停用此功能，請將 use_gzip_http_content_encoding = false 新增至 CloudWatch Logs 代理程式組態檔案的 [general] 區段，然後重新啟動代理程式。

> **Note**
>
> 此設定僅適用於 awscli-cwlogs 1.3.3 及更新版本。

log_group_name

指定目的地日誌群組。如果日誌群組尚未存在，將會自動建立。日誌群組的名稱長度可介於 1 到 512 個字元之間。可用的字元為 a-z、A-Z、0-9、「_」(底線)、「-」(連字號)、「/」(正斜線) 和「.」(句點)。

log_stream_name

指定目的地日誌串流。您可以使用常值字串、預先定義的變數 ({instance_id}、{hostname}、{ip_address}) 或這兩者的組合，以定義日誌串流名稱。如果日誌串流尚未存在，將會自動建立。

datetime_format

指定從日誌擷取時間戳記的方式。此時間戳記用於擷取日誌事件及產生指標。若未提供 datetime_format，目前時間將用於每個日誌事件。如果提供的 datetime_format 值對於指定的日誌訊息而言是無效的，這時將使用最近一次所含時間戳記成功剖析之日誌事件的時間戳記。如果沒有之前的日誌事件，將使用目前時間。

以下列出常見的 datetime_format 代碼。您也可以使用任何 Python 支援的 datetime_format 代碼、datetime.strptime()。亦支援時區位移 (%z)，雖然 Python 3.2 之前版本並不支援，[+-] HHMM 無需冒號 (:)。如需詳細資訊，請參閱 strftime () 和 strptime () 行為。

%y：年，不包含以填充零之十進位表示的世紀數字。00、01、…、99

%Y：年，包含以十進位表示的世紀數字。1970、1988、2001、2013

%b：月，當地的縮寫名稱。Jan、Feb、…、Dec (en_US)；

%B：月，當地的完整名稱。January、February、…、December (en_US)；

%m：月，填充零的十進位數字。01、02、…、12

%d：日，填充零的十進位數字。01、02、…、31

%H：小時 (24 小時制)，填充零的十進位數字。00、01、…、23

%I：小時 (12 小時制)，填充零的十進位數字。01、02、…、12

%p：相當於當地的 AM 或 PM。

%M：分鐘，填充零的十進位數字。00、01、…、59

%S：秒鐘，填充零的十進位數字。00、01、…、59

%f：微秒，十進位數字，左側填充零。000000、…、999999

%z：UTC 位移，格式為 +HHMM 或 -HHMM。+0000、-0400、+1030

範例格式：

```
Syslog: '%b %d %H:%M:%S', e.g. Jan 23 20:59:29

Log4j: '%d %b %Y %H:%M:%S', e.g. 24 Jan 2014 05:00:00

ISO8601: '%Y-%m-%dT%H:%M:%S%z', e.g. 2014-02-20T05:20:20+0000
```

time_zone

指定日誌事件時間戳記的時區。支援的兩個值為 UTC 和 LOCAL。如果無法依據 datetime_format 推斷時區，預設值為 LOCAL。

file

指定您要推送到 CloudWatch Logs 的日誌檔。檔案可指向特定檔案或多個檔案 (使用萬用字元，例如 /var/log/system.log*)。根據檔案的修改時間，只會將最新的檔案推送到 CloudWatch Logs。我們建議您使用萬用字元來指定一系列的相同類型的檔案，例如 access_log.2014-06-01-01、access_log.2014-06-01-02，以此類推，但不適用於多種種類的檔案，例如 access_log_80 和 access_log_443。若要指定多種種類的檔案，可將另一個日誌串流新增至組態檔案，讓每個種類的日誌檔進入不同的日誌串流。不支援壓縮檔案。

file_fingerprint_lines

指定用於識別檔案的行範圍。有效值是一個數字或兩個以破折號分隔的數字，例如「1」、「2-5」。預設值為「1」，因此第一行用於計算指紋。除非所有指定的行皆可用，否則不會將指紋行傳送到 CloudWatch Logs。

multi_line_start_pattern

指定用於識別日誌訊息開始處的模式。日誌訊息是由符合模式的一列及不符合模式的任何幾列所組成。有效值為規則表達式或 {datetime_format}。使用 {datetime_format} 時，應指定 datetime_format 選項。預設值為「^ [^\s]」，因此開頭使用非空白字元的任何列皆可結束之前的日誌訊息，並開始新的日誌訊息。

initial_position

指定開始讀取資料 (start_of_file 或 end_of_file) 的位置。預設值為 start_of_file。只有在日誌串流沒有狀態時才會使用它。

編碼

指定日誌檔的編碼，以便正確讀取檔案。預設值為 utf_8。這裡可以使用 Python codecs.decode () 支援的編碼。

> ⚠ Warning
>
> 指定不正確的編碼可能導致資料遺失，因為無法解碼的字元會被替換為一些其他的字元。

以下是一些常見的編碼：

```
ascii, big5, big5hkscs, cp037, cp424, cp437, cp500, cp720, cp737,
cp775, cp850, cp852, cp855, cp856, cp857, cp858, cp860, cp861, cp862,
cp863, cp864, cp865, cp866, cp869, cp874, cp875, cp932, cp949, cp950,
cp1006, cp1026, cp1140, cp1250, cp1251, cp1252, cp1253, cp1254, cp1255,
cp1256, cp1257, cp1258, euc_jp, euc_jis_2004, euc_jisx0213, euc_kr,
gb2312, gbk, gb18030, hz, iso2022_jp, iso2022_jp_1, iso2022_jp_2,
iso2022_jp_2004, iso2022_jp_3, iso2022_jp_ext, iso2022_kr, latin_1,
iso8859_2, iso8859_3, iso8859_4, iso8859_5, iso8859_6, iso8859_7,
iso8859_8, iso8859_9, iso8859_10, iso8859_13, iso8859_14, iso8859_15,
iso8859_16, johab, koi8_r, koi8_u, mac_cyrillic, mac_greek, mac_iceland,
mac_latin2, mac_roman, mac_turkish, ptcp154, shift_jis, shift_jis_2004,
```

```
shift_jisx0213, utf_32, utf_32_be, utf_32_le, utf_16, utf_16_be,
utf_16_le, utf_7, utf_8, utf_8_sig
```

buffer_duration

指定日誌事件的批次處理的持續時間。最小值為 5000ms，預設值為 5000ms。

batch_count

指定批次中的日誌事件最大數量，最多可達 10,000。預設值為 10000。

batch_size

指定批次中的日誌事件最大大小，以位元組為單位，最多可達 1048576 位元組。預設值為 1048576 位元組。這個大小的計算方式是以所有 UTF-8 事件訊息，加上每個記錄事件 26 個位元組。

透過 HTTP 代理使用 CloudWatch Logs 代理程式

您可以透過 HTTP 代理來使用 CloudWatch Logs 代理程式。

> **Note**
>
> awslogs-agent-setup.py 1.3.8 或更新版本支援 HTTP 代理。

透過 HTTP 代理使用 CloudWatch Logs 代理程式

1. 執行以下任意一項：

   a. 針對新安裝的 CloudWatch Logs 代理程式，請執行下列命令：

   ```
   curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
   setup.py -O
   ```

   ```
   sudo python awslogs-agent-setup.py --region us-east-1 --http-proxy http://your/
   proxy --https-proxy http://your/proxy --no-proxy 169.254.169.254
   ```

   為了維持存取 EC2 執行個體上的 Amazon EC2 中繼資料服務，請使用 --no-proxy 169.254.169.254 (建議)。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的執行個體中繼資料和使用者資料。

在 `http-proxy` 和 `https-proxy` 的數值中，您必須指定整個 URL。

b. 針對現有已安裝的 CloudWatch Logs 代理程式，請編輯 /var/awslogs/etc/proxy.conf，並新增您的代理：

```
HTTP_PROXY=
HTTPS_PROXY=
NO_PROXY=
```

2. 重新啟動代理程式，讓變更生效：

```
sudo service awslogs restart
```

如果您使用的是 Amazon Linux 2，請使用下列命令來重新啟動代理程式：

```
sudo service awslogsd restart
```

分隔 CloudWatch Logs 代理程式組態檔案

如果您使用的是 awslogs-agent-setup.py 1.3.8 或更新版本及 awscli-cwlogs 1.3.3 或更新版本，您可以在 /var/awslogs/etc/config/ 目錄中建立額外的組態檔案，為各種元件獨立匯入不同的串流組態。CloudWatch Logs 代理程式啟動時會包含這些額外組態檔案中的任何串流組態。[general] 區段中的組態屬性必須在主要組態檔案 (/var/awslogs/etc/awslogs.conf) 中定義，並且在 /var/awslogs/etc/config/ 中的任何額外的組態檔案中被忽略。

如果您因為使用 rpm 安裝代理程式，因此沒有 /var/awslogs/etc/config/ 目錄，您可以改為使用 /etc/awslogs/config/ 目錄。

重新啟動代理程式，讓變更生效：

```
sudo service awslogs restart
```

如果您使用的是 Amazon Linux 2，請使用下列命令來重新啟動代理程式：

```
sudo service awslogsd restart
```

CloudWatch Logs 代理程式常見問答集

支援哪些種類的檔案輪換？

支援以下檔案輪換機制：

- 以數值尾碼重新命名現有的日誌檔，然後重新建立原始的空日誌檔。例如，/var/log/syslog.log 重新命名為 /var/log/syslog.log.1. 如果 /var/log/syslog.log.1 從之前的輪換就已存在，它將會重新命名為 /var/log/syslog.log.2。
- 在建立副本之後，截斷已備妥的原始日誌檔。例如，/var/log/syslog.log 將會複製到 /var/log/syslog.log.1，/var/log/syslog.log 將被截斷。在這種情況下，資料可能會遺失，因此請留意使用此檔案輪換機制。
- 使用與舊檔案相同的通用模式建立新檔案。例如，保留 /var/log/syslog.log.2014-01-01，並建立 /var/log/syslog.log.2014-01-02。

檔案的指紋 (來源 ID) 的計算方式是雜湊日誌串流金鑰與檔案的第一行內容。若要覆寫此行為，可使用 file_fingerprint_lines 選項。發生檔案輪換時，新的檔案應該會有新的內容，舊的檔案不應該有附加的內容；代理程式會在完成讀取舊檔案之後推送新的檔案。

我如何判斷我使用的代理程式是哪個版本？

如果是使用設定指令碼來安裝 CloudWatch Logs 代理程式，您可以使用 /var/awslogs/bin/awslogs-version.sh 來檢查您使用的代理程式是哪個版本。它會列印出代理程式的版本及其主要相依性。如果是使用 yum 來安裝 CloudWatch Logs 代理程式，您可以使用 "yum info awslogs" 和 "yum info aws-cli-plugin-cloudwatch-logs" 來檢查 CloudWatch Logs 代理程式和外掛程式的版本。

日誌項目如何轉換為日誌事件？

日誌事件包含兩個屬性：事件發生時的時間戳記，以及原始日誌訊息。依據預設，開頭使用非空白字元的任何列皆可結束之前的日誌訊息 (如果有的話)，並開始新的日誌訊息。若要覆寫此行為，可以使用 multi_line_start_pattern，符合此模式的任何列都會開始新的日誌訊息。此模式可以是任何 regex 或「{datetime_format}」。例如，如果每個日誌訊息的第一行包含時間戳記，例如 '2014-01-02T13:13:01Z'，則 multi_line_start_pattern 可設定為 '\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}Z'。為了簡化組態，如果已指定 datetime_format option，您可以使用「{datetime_format}」變數。以相同的範例而言，如果 datetime_format 設為 '%Y-%m-%dT%H:%M:%S%z'，則 multi_line_start_pattern 可以只是「{datetime_format}」。

若未提供 datetime_format，目前時間將用於每個日誌事件。如果提供的 datetime_format 對於指定的日誌訊息而言是無效的，將使用最後一個成功剖析時間戳記的日誌事件的時間戳記。如果沒有之前的日誌事件，將使用目前時間。當日誌事件回退至目前時間或之前的日誌事件時間，將會記錄警告訊息。

時間戳記用於擷取日誌事件並產生指標，因此如果您指定錯誤的格式，可能會導致無法擷取日誌事件，並且會產生錯誤的指標。

日誌事件會如何進行批次處理？

當下列任何一個條件成立時，批次將變滿並予以發佈：

1. 從新增第一個日誌事件以來，已經過 buffer_duration 的時間長度。
2. 已累積小於 batch_size 的日誌事件，但新增超過 batch_size 的新日誌事件。
3. 達到 batch_count 的日誌事件數量。
4. 此批次的日誌事件未持續超過 24 小時，但新增新日誌事件過程超過 24 小時限制。

什麼會導致日誌項目、日誌事件，或批次遭到略過或截斷？

若要遵循 `PutLogEvents` 操作的限制，以下問題可能會導致日誌事件或批次進行被略過。

> **Note**
>
> 略過資料時，CloudWatch Logs 代理程式會將警告寫入日誌。

1. 如果日誌事件的大小超過 256 KB，將會完全略過該日誌事件。
2. 如果日誌事件的時間戳記超過未來 2 小時，將會略過該日誌事件。
3. 如果日誌事件的時間戳記超過過去 14 天，將會略過該日誌事件。
4. 如有任何日誌事件超過日誌群組的保留期間，將會略過整個批次。
5. 如果在單一 `PutLogEvents` 請求中的日誌事件批次持續超過 24 小時，則 `PutLogEvents` 操作會失敗。

停用代理程式是否會造成資料遺失/重複？

只要有狀態檔案，而且從上次執行之後未發生檔案輪換，就不會造成資料遺失/重複。CloudWatch Logs 代理程式可以從停止的位置開始，並繼續推送日誌資料。

我是否可以從相同或不同的主機，將不同的日誌檔指向相同的日誌串流？

不支援設定多個日誌來源，將資料傳送到單個日誌串流。

代理程式發出哪些 API 呼叫 (或我應該將哪些動作新增至 IAM 政策)？

CloudWatch Logs 代理程式需要執行 `CreateLogGroup`、`CreateLogStream`、`DescribeLogGrooupd`、`DescribeLogStreamsPutLogEvents`和 `PutRetentionPolicy`動作的許可。如果您使用的是最新的代理程式，則不需要 `DescribeLogStreams`。請參閱以下 IAM 政策範例。

JSON

```
{
"Version":"2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "logs:CreateLogGroup",
      "logs:CreateLogStream",
      "logs:PutLogEvents",
      "logs:DescribeLogStreams",
      "logs:DescribeLogGroups",
      "logs:PutRetentionPolicy"
    ],
    "Resource": [
      "arn:aws:logs:*:*:*"
    ]
  }
 ]
}
```

我不希望 CloudWatch Logs 代理程式自動建立日誌群組或日誌串流。我要如何防止代理程式重新建立日誌群組與日誌串流？

您可以在 IAM 政策中限制代理程式只能執行以下操作：`DescribeLogStreams`、`PutLogEvents`。

從代理程式撤銷 `CreateLogGroup` 和 `CreateLogStream` 許可前，請務必建立您要讓代理程式使用的日誌群組和日誌串流。日誌代理程式無法在您已建立的日誌群組中建立日誌串流，除非其擁有 `CreateLogGroup` 和 `CreateLogStream` 許可。

進行故障排除時，我應該查看哪些日誌？

代理程式安裝日誌位於 `/var/log/awslogs-agent-setup.log`，代理程式日誌位於 `/var/log/awslogs.log`。

快速入門：使用 CloudFormation 來開始使用 CloudWatch Logs

AWS CloudFormation 可讓您以 JSON 格式描述和佈建 AWS 資源。此方法的優點包括能夠以單一單位管理 AWS 資源集合，以及輕鬆跨區域複寫您的 AWS 資源。

當您 AWS 使用 佈建 時 CloudFormation，您可以建立範本來描述 AWS 要使用的資源。以下範例是一種範本程式碼片段，其會建立一個日誌群組和指標篩選條件以計算 404 發生次數並將此計數傳送至日誌群組。

```
"WebServerLogGroup": {
    "Type": "AWS::Logs::LogGroup",
    "Properties": {
        "RetentionInDays": 7
    }
},

"404MetricFilter": {
    "Type": "AWS::Logs::MetricFilter",
    "Properties": {
        "LogGroupName": {
            "Ref": "WebServerLogGroup"
        },
        "FilterPattern": "[ip, identity, user_id, timestamp, request, status_code =
 404, size, ...]",
        "MetricTransformations": [
            {
                "MetricValue": "1",
                "MetricNamespace": "test/404s",
                "MetricName": "test404Count"
            }
        ]
    }
}
```

這是一個基本的範例。您可以使用 設定更豐富的 CloudWatch Logs 部署 CloudFormation。如需範本範例的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 Amazon CloudWatch Logs 範本程式碼片段。如需有關入門的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 AWS CloudFormation入門。

透過 HTTP 端點記錄擷取

Amazon CloudWatch Logs 提供 HTTP 端點，可讓您使用簡單的 HTTP POST 請求將日誌直接傳送至 CloudWatch Logs。這些端點同時支援 SigV4 和承載字符身分驗證。

> **Important**
>
> 我們建議對可以整合 AWS SDK 的所有生產工作負載使用 SigV4 身分驗證。SigV4 使用短期登入資料，並提供最強大的安全狀態。承載字符 (API 金鑰） 身分驗證適用於 SigV4 不可行的情況，例如不支援 AWS SDK 整合的第三方日誌轉送器。如需詳細資訊，請參閱《IAM 使用者指南》中的長期存取金鑰的替代方案。

CloudWatch Logs 支援下列 HTTP 擷取端點：

Endpoint	路徑	內容類型	格式
OpenTelemetry Logs	`/v1/logs`	`application/json` 或 `application/x-protobuf`	OTLP JSON 或 Protobuf
HLC Logs	`/services/collector/event`	`application/json`	HLC 格式
ND-JSON Logs	`/ingest/bulk`	`application/json` 或 `application/x-ndjson`	換行分隔的 JSON
Structured JSON Logs	`/ingest/json`	`application/json`	JSON 物件或陣列

常見行為

所有 HTTP 擷取端點都共用下列行為：

身分驗證

所有端點都支援 SigV4 和承載字符身分驗證：

- SigV4 （建議） – 標準 AWS 簽章第 4 版簽署。每當您的應用程式或基礎設施支援 AWS SDK 或 可以簽署請求時，請使用 SigV4。SigV4 使用短期憑證，是最安全的身分驗證方法。
- 承載字符 – 使用 `Authorization: Bearer <ACWL token>`標頭。
  - 權杖必須是有效的 ACWL 承載權杖。如需設定說明，請參閱 the section called “承載字符身分驗證”。
  - 需要 `logs:PutLogEvents`和 `logs:CallWithBearerToken` IAM 許可。

日誌群組和日誌串流

- 透過標頭提供： `x-aws-log-group`和 `x-aws-log-stream`
- 除了 `?logGroup=<name>&logStream=<name>` OTLP 之外，所有端點也支援查詢參數。
- 您無法對相同的參數同時使用查詢參數和標頭。
- 日誌群組和日誌串流都是必要的。

回應

- 成功：`HTTP 200`使用內文 `{}`
- 驗證錯誤： `HTTP 400`
- 驗證失敗： `HTTP 401`

設定承載字符身分驗證

在使用承載字符身分驗證與任何 HTTP 擷取端點傳送日誌之前，您需要：

- 使用 CloudWatch Logs 許可建立 IAM 使用者
- 產生服務特定的登入資料 （承載字符）
- 建立日誌群組和日誌串流
- 在日誌群組上啟用承載字符身分驗證

> **Important**
>
> 我們建議針對所有工作負載使用 SigV4 身分驗證搭配短期登入資料，以達成此目的。SigV4 提供最強大的安全狀態。限制在短期憑證型身分驗證不可行的情況下使用 API 金鑰 （承載字

符）。當您準備好將 CloudWatch Logs 納入具有更高安全需求的應用程式時，您應該切換到短期憑證。如需詳細資訊，請參閱《IAM 使用者指南》中的長期存取金鑰的替代方案。

選項 1：使用 AWS 主控台快速開始使用

AWS 管理主控台提供簡化的工作流程，為 HTTP 端點存取產生 API 金鑰。

使用主控台設定 HTTP 端點存取

1. 登入 AWS 管理主控台。
2. 導覽至 CloudWatch > 設定 > 日誌。
3. 在 API 金鑰區段中，選擇產生 API 金鑰。
4. 對於 API 金鑰過期，請執行下列其中一項操作：
   - 選取 1、5、30、90 或 365 天的 API 金鑰過期持續時間。
   - 選擇自訂持續時間以指定自訂 API 金鑰過期日期。
   - 選取永不過期（不建議）。
5. 選擇產生 API 金鑰。

   主控台會自動：
   - 建立具有適當許可的新 IAM 使用者
   - 連接 CloudWatchLogsAPIKeyAccess 受管政策（包含 `logs:PutLogEvents`和 `logs:CallWithBearerToken`許可）
   - 產生服務特定的登入資料 (API 金鑰）
6. 複製並安全地儲存顯示的登入資料：
   - API 金鑰 ID（服務特定的登入資料 ID)
   - API 金鑰秘密（承載字符）

   > **Important**
   > 立即儲存 API 金鑰秘密。它稍後無法擷取。如果遺失，您將需要產生新的 API 金鑰。
7. 建立將存放日誌的日誌群組和日誌串流：

```
# Create the log group
aws logs create-log-group \
    --log-group-name /aws/hlc-logs/my-application \
    --region us-east-1

# Create the log stream
aws logs create-log-stream \
    --log-group-name /aws/hlc-logs/my-application \
    --log-stream-name application-stream-001 \
    --region us-east-1
```

8. 在日誌群組上啟用承載字符身分驗證：

```
aws logs put-bearer-token-authentication \
    --log-group-identifier /aws/hlc-logs/my-application \
    --bearer-token-authentication-enabled \
    --region us-east-1
```

驗證組態：

```
aws logs describe-log-groups \
    --log-group-name-prefix /aws/hlc-logs/my-application \
    --region us-east-1
```

包含的許可：自動建立的 IAM 使用者將具有下列許可：

- `logs:PutLogEvents` – 將日誌事件傳送至 CloudWatch Logs
- `logs:CallWithBearerToken` – 使用承載字符進行驗證
- `kms:Describe*`、`kms:GenerateDataKey*`、 `kms:Decrypt` – 存取 KMS 加密日誌群組（具有限制為日誌服務的條件）

選項 2：手動設定

如果您偏好對 IAM 組態進行更多控制，或需要自訂許可，您可以手動設定 HTTP 端點存取。

步驟 1：建立 IAM 使用者

建立將用於日誌擷取的 IAM 使用者：

1. 登入 AWS 管理主控台並導覽至 IAM。
2. 在左側導覽窗格中，選擇 Users (使用者)。
3. 選擇 Create user (建立使用者)。
4. 輸入使用者名稱（例如 `cloudwatch-logs-hlc-user`)。
5. 選擇下一步。
6. 連接下列其中一個 IAM 政策：

   選項 A：使用 受管政策（建議）

   連接 CloudWatchLogsAPIKeyAccess 受管政策。

   選項 B：建立自訂政策

   建立並連接下列 IAM 政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "LogsAPIs",
            "Effect": "Allow",
            "Action": [
                "logs:CallWithBearerToken",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "KMSAPIs",
            "Effect": "Allow",
            "Action": [
                "kms:Describe*",
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": [
                        "logs.*.amazonaws.com"
                    ]
                }
```

```
            },
            "Resource": "arn:aws:kms:*:*:key/*"
        }
    ]
}
```

7. 選擇下一步，然後選擇建立使用者。

> **Note**
>
> 如果您計劃將日誌傳送至 KMS 加密的日誌群組，則需要 KMS 許可。條件限制 KMS 只能存取透過 CloudWatch Logs 服務使用的金鑰。

步驟 2：產生服務特定的登入資料 (API 金鑰)

使用 CreateServiceSpecificCredential API 產生 CloudWatch Logs API 金鑰。您也可以使用 create-service-specific-credential CLI 命令。對於憑證存留期，您可以指定介於 1–36600 天之間的值。如果未指定憑證存留期，則 API 金鑰將不會過期。

若要產生過期 30 天的 API 金鑰：

```
aws iam create-service-specific-credential \
    --user-name cloudwatch-logs-hlc-user \
    --service-name logs.amazonaws.com \
    --credential-age-days 30
```

回應是 ServiceSpecificCredential 物件。`ServiceCredentialSecret` 值是您的 CloudWatch Logs API 金鑰 （承載字符）。

> **Important**
>
> 請妥善儲存 `ServiceCredentialSecret` 值，因為您之後將無法再擷取它。如果遺失，您將需要產生新的 API 金鑰。

步驟 3：建立日誌群組和日誌串流

建立將存放日誌的日誌群組和日誌串流：

```
# Create the log group
aws logs create-log-group \
    --log-group-name /aws/hlc-logs/my-application \
    --region us-east-1

# Create the log stream
aws logs create-log-stream \
    --log-group-name /aws/hlc-logs/my-application \
    --log-stream-name application-stream-001 \
    --region us-east-1
```

步驟 4：啟用承載字符身分驗證

在日誌群組上啟用承載字符身分驗證：

```
aws logs put-bearer-token-authentication \
    --log-group-identifier /aws/hlc-logs/my-application \
    --bearer-token-authentication-enabled \
    --region us-east-1
```

驗證組態：

```
aws logs describe-log-groups \
    --log-group-name-prefix /aws/hlc-logs/my-application \
    --region us-east-1
```

控制產生和使用 CloudWatch Logs API 金鑰的許可

CloudWatch Logs API 金鑰的產生和使用由 CloudWatch Logs 和 IAM 服務中的動作和條件金鑰控制。

控制 CloudWatch Logs API 金鑰的產生

iam：CreateServiceSpecificCredential 動作會控制產生服務特定的金鑰（例如 CloudWatch Logs API 金鑰）。您可以將此動作的範圍限定為 IAM 使用者，以此限制可以為其產生金鑰的使用者。

您可以使用下列條件索引鍵，對 `iam:CreateServiceSpecificCredential` 動作的許可施加條件：

- iam:ServiceSpecificCredentialAgeDays – 可讓您在條件中指定金鑰的過期時間，以天為單位。例如，您可以使用此條件索引鍵，僅允許建立有效期為 90 天的 API 金鑰。

- iam:ServiceSpecificCredentialServiceName – 可讓您在條件中指定服務的名稱。例如，您可以使用此條件金鑰來僅允許為 CloudWatch Logs 建立 API 金鑰，而非其他 服務。

控制 CloudWatch Logs API 金鑰的使用

`logs:CallWithBearerToken` 動作控制 CloudWatch Logs API 金鑰的使用。若要防止身分使用 CloudWatch Logs API 金鑰，請將拒絕`logs:CallWithBearerToken`動作的政策連接至與金鑰相關聯的 IAM 使用者。

政策範例

防止身分產生和使用 CloudWatch Logs API 金鑰

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyCWLAPIKeys",
            "Effect": "Deny",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "logs:CallWithBearerToken"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Warning**
>
> 此政策將防止為所有支援建立服務特定登入資料的 AWS 服務建立登入資料。如需詳細資訊，請參閱 IAM 使用者的服務特定憑證。

防止身分使用 CloudWatch Logs API 金鑰

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Deny",
            "Action": "logs:CallWithBearerToken",
            "Resource": "*"
        }
    ]
}
```

只有在 CloudWatch Logs 金鑰在 90 天內過期時，才允許建立金鑰

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::123456789012:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName": "logs.amazonaws.com"
                },
                "NumericLessThanEquals": {
                    "iam:ServiceSpecificCredentialAgeDays": "90"
                }
            }
        }
    ]
}
```

輪換 API 金鑰

定期輪換您的 API 金鑰可降低未經授權的存取風險。我們建議您建立符合您組織安全政策的輪換排程。

輪換程序

若要在不中斷日誌交付的情況下輪換 API 金鑰，請遵循下列程序：

1. 為 IAM 使用者建立新的（次要）登入資料：

```
aws iam create-service-specific-credential \
    --user-name cloudwatch-logs-hlc-user \
    --service-name logs.amazonaws.com \
```

```
    --credential-age-days 90
```

2. （選用）將新登入資料存放在 中， AWS Secrets Manager 以安全擷取和自動輪換。
3. 將新登入資料匯入廠商的入口網站，或更新您的應用程式組態以使用新的 API 金鑰。
4. 將原始登入資料設定為非作用中：

```
aws iam update-service-specific-credential \
    --user-name cloudwatch-logs-hlc-user \
    --service-specific-credential-id ACCA1234EXAMPLE1234 \
    --status Inactive
```

5. 在 CloudWatch `IncomingBytes` 中監控日誌群組的指標，確認日誌交付未受到影響。如需詳細資訊，請參閱使用 CloudWatch 指標監控使用量。
6. 使用新金鑰確認成功交付後，請刪除先前的登入資料：

```
aws iam delete-service-specific-credential \
    --service-specific-credential-id ACCA1234EXAMPLE1234
```

監控金鑰過期

若要檢查現有 API 金鑰的建立日期和狀態，請使用 list-service-specific-credentials 命令：

```
aws iam list-service-specific-credentials \
    --user-name cloudwatch-logs-hlc-user \
    --service-name logs.amazonaws.com
```

回應包含每個登入`Status`資料的 `CreateDate`和 。使用此資訊來識別即將到期或已處於作用中狀態超過輪換政策允許的金鑰。

回應遭入侵的 API 金鑰

如果您懷疑 API 金鑰已洩露，請立即採取下列步驟：

1. 立即停用金鑰以防止進一步未經授權的使用：

```
aws iam update-service-specific-credential \
    --user-name cloudwatch-logs-hlc-user \
    --service-specific-credential-id ACCA1234EXAMPLE1234 \
    --status Inactive
```

2. 檢閱 CloudTrail 日誌以判斷未經授權的存取範圍。如需如何啟用 API 金鑰用量的稽核the section called “使用 CloudTrail 記錄 API 金鑰用量”，請參閱 。
3. 依照中所述的輪換程序建立替換金鑰the section called “輪換程序”。
4. 取代完成後刪除遭入侵的金鑰：

```
aws iam delete-service-specific-credential \
    --service-specific-credential-id ACCA1234EXAMPLE1234
```

5. 如果您需要在調查時立即封鎖 IAM 使用者的所有承載字符存取，請連接拒絕政策：

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "logs:CallWithBearerToken",
        "Resource": "*"
    }
}
```

> **Note**
>
> 若要透過 API 執行這些動作，您必須使用 AWS 登入資料進行驗證，而不是使用 CloudWatch Logs API 金鑰進行驗證。

您也可以使用下列 IAM API 操作來管理遭到入侵的金鑰：

- ResetServiceSpecificCredential – 重設金鑰以產生新密碼，而不刪除登入資料。金鑰不得已過期。

API 金鑰的安全最佳實務

遵循這些最佳實務來保護您的 CloudWatch Logs API 金鑰：

- 切勿在原始程式碼中嵌入 API 金鑰。請勿在應用程式程式碼中硬式編碼 API 金鑰，或將其遞交至版本控制系統。如果金鑰意外遞交到公有儲存庫， AWS 自動化掃描可能會標記它，您應該立即輪換金鑰。
- 使用秘密管理員。將 API 金鑰存放在 AWS Secrets Manager或同等的秘密管理解決方案中。這可啟用集中式存取控制、稽核記錄和自動輪換。

- 在所有金鑰上設定過期。建立 API 金鑰時，請務必指定`--credential-age-days`值。若要在整個組織中強制執行最長金鑰生命週期，請使用 IAM `iam:ServiceSpecificCredentialAgeDays` 條件金鑰。如需範例，請參閱 the section called “只有在 CloudWatch Logs 金鑰在 90 天內過期時，才允許建立金鑰”。
- 套用最低權限許可。僅將 IAM 使用者的許可範圍限定為所需的日誌群組和動作。使用 受管 CloudWatchLogsAPIKeyAccess 政策作為起點，並視需要進一步限制。
- 啟用 CloudTrail 記錄。透過啟用 的 CloudTrail 資料事件來稽核 API 金鑰用量`AWS::Logs::LogGroupAuthorization`。請參閱 the section called “使用 CloudTrail 記錄 API 金鑰用量”。
- 使用 IAM Access Analyzer 監控 。使用 IAM Access Analyzer 識別未使用的登入資料，以及與您的 API 金鑰 IAM 使用者相關聯的過度寬鬆政策。
- 定期輪換金鑰。建立輪換排程並遵循中所述的程序the section called “輪換 API 金鑰”。

使用 CloudTrail 記錄 API 金鑰用量

您可以使用 AWS CloudTrail 記錄 CloudWatch Logs API 金鑰用量的資料事件。CloudWatch Logs 會發出`CallWithBearerToken`呼叫`AWS::Logs::LogGroupAuthorization`的資料事件，讓您稽核何時及如何使用 API 金鑰傳送日誌。

若要為 CloudWatch Logs API 金鑰用量啟用 CloudTrail 記錄： CloudWatch

> **Note**
>
> 您為線索指定的 S3 儲存貯體必須具有允許 CloudTrail 將日誌檔案寫入其中的儲存貯體政策。如需詳細資訊，請參閱 CloudTrail 的 Amazon S3 儲存貯體政策。

1. 建立線索：

```
aws cloudtrail create-trail \
    --name cloudwatch-logs-api-key-audit \
    --s3-bucket-name my-cloudtrail-bucket \
    --region us-east-1
```

2. 設定進階事件選取器以擷取 CloudWatch Logs 日誌群組授權事件：

```
aws cloudtrail put-event-selectors \
    --region us-east-1 \
```

```
    --trail-name cloudwatch-logs-api-key-audit \
    --advanced-event-selectors '[{
        "Name": "CloudWatch Logs API key authorization events",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["Data"] },
            { "Field": "resources.type", "Equals":
 ["AWS::Logs::LogGroupAuthorization"] }
        ]
    }]'
```

3. 開始追蹤記錄：

```
aws cloudtrail start-logging \
    --name cloudwatch-logs-api-key-audit \
    --region us-east-1
```

使用 OTLP 端點傳送日誌 (OpenTelemetry Logs)

OpenTelemetry Logs 端點 (`/v1/logs`) 接受 JSON 或 Protobuf 編碼中的 OpenTelemetry Protocol (OTLP) 日誌資料。如需 OTLP 端點的詳細資訊，包括組態和用量，請參閱使用 OpenTelemetry 將指標和追蹤傳送至 CloudWatch。

如果您使用承載字符身分驗證，請先完成 中的設定步驟，the section called “承載字符身分驗證”再繼續。

要求格式

- 方法： `POST`
- Content-Type： `application/json`或 `application/x-protobuf`
- 日誌群組：僅限 `x-aws-log-group` 標頭（不支援查詢參數）
- 日誌串流：`x-aws-log-stream`標頭

範例請求

```
curl -X POST "https://logs.<region>.amazonaws.com/v1/logs" \
  -H "Authorization: Bearer ACWL<token>" \
  -H "Content-Type: application/json" \
  -H "x-aws-log-group: MyLogGroup" \
```

```
  -H "x-aws-log-stream: MyLogStream" \
  -d '{
  "resourceLogs": [
    {
      "resource": {
        "attributes": [
          {
            "key": "service.name",
            "value": { "stringValue": "my-service" }
          }
        ]
      },
      "scopeLogs": [
        {
          "scope": {
            "name": "my-library",
            "version": "1.0.0"
          },
          "logRecords": [
            {
              "timeUnixNano": "1741900000000000000",
              "severityNumber": 9,
              "severityText": "INFO",
              "body": {
                "stringValue": "User logged in successfully"
              },
              "attributes": [
                {
                  "key": "user.id",
                  "value": { "stringValue": "12345" }
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}'
```

回應

成功（接受所有事件）：

```
HTTP 200 OK
{}
```

部分成功 （ 部分事件遭到拒絕 ）：

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1,
 \"expiredLogEventCount\": 1}"
  }
}
```

當請求 Content-Type 為 時`application/x-protobuf`，回應會以具有相同欄位的序列化 `ExportLogsServiceResponse` protobuf 訊息傳回。

OTLP 特定行為

下列行為是 OTLP 端點特有的行為，不會出現在其他 HTTP 擷取端點上：

- Retry-After 標頭 – 包含在 503 和 429 回應上，指出用戶端何時應重試。

使用 HLC 端點傳送日誌 (HLC 日誌)

HLC Logs 端點 (`/services/collector/event`) 是以 HTTP Log Collector (HLC) 格式為基礎。

如果您使用承載字符身分驗證，請先完成 中的設定步驟，the section called “承載字符身分驗證”再繼續。

輸入模式

每個事件都是具有必要`"event"`欄位的 JSON 物件。選用中繼資料欄位：`"time"`、`"host"`、`"source"`、`"sourcetype"`、`"index"`。

單一事件：

```
{"event":"Hello world!","time":1486683865.0}
```

事件的 JSON 陣列：

```
[
  {"event":"msg1","time":1486683865.0},
  {"event":"msg2","time":1486683866.0}
]
```

串連/批次事件（無陣列包裝函式）：

```
{"event":"msg1","time":1486683865.0}{"event":"msg2","time":1486683866.0}
```

事件欄位（必要）

欄位為必要"event"欄位。其值可以是任何 JSON 類型：

```
{"event":"a string message"}
{"event":{"message":"structured data","severity":"INFO"}}
{"event":42}
{"event":true}
```

無 "event" 欄位的物件會無提示地略過：

```
{"message":"this is skipped — no event field"}
```

時間欄位（選用）

"time" 欄位以 epoch 秒（非毫秒）為單位，選用小數表示次秒精確度。

格式	範例	解譯為
Float	"time":1486683865.500	1486683865500 毫秒
Integer	"time":1486683865	1486683865000 毫秒
字串（浮點數）	"time":"1486683865.500"	1486683865500 毫秒
字串（整數）	"time":"1486683865"	1486683865000 毫秒
缺少	（無時間欄位）	伺服器目前時間
無效	"time":"invalid"	伺服器目前時間

內容類型

僅接受 `application/json` 。

接受的 JSON 值類型

最上層類型	Behavior (行為)
具有 的物件 `"event"`	已接受
沒有 的物件 `"event"`	略過
物件的陣列	個別處理的每個元素
串連物件	個別處理的每個物件
基本 （ 字串、數字、布林值、 null)	略過

端點格式

HLC 端點 URL 遵循此格式：

```
https://logs.<region>.amazonaws.com/services/collector/event?
logGroup=<name>&logStream=<name>[&entityName=<name>&entityEnvironment=<environment>]
```

必要參數：

- `<region>` – AWS Region （ 例如，`us-east-1`、`eu-west-1`)
- `logGroup` – URL 編碼的日誌群組名稱
- `logStream` – URL 編碼的日誌串流名稱

選用參數：

您可以選擇性地將日誌事件與`Service`實體建立關聯，方法是包含下列查詢參數。由於透過 HLC 端點傳送的日誌是自訂遙測，因此不會自動與實體建立關聯。透過提供這些參數，CloudWatch Logs 會建立將 `KeyAttributes.Type` 設定為 的實體，`Service`並將其與您的日誌事件建立關聯。這可讓 CloudWatch 中的探索相關功能將這些日誌與來自相同服務的其他遙測 （ 指標、追蹤和日誌） 建立關

聯，讓您更輕鬆地疑難排解和監控不同訊號類型的應用程式。如需實體和相關遙測的詳細資訊，請參閱將相關資訊新增至自訂遙測。

- entityName – 要與日誌事件建立關聯的服務實體名稱。此值會儲存為實體 KeyAttributes.Name (例如 my-application或 api.myservice.com)。
- entityEnvironment – 託管服務或其所屬的環境。此值會儲存為實體 KeyAttributes.Environment (例如 production、 ec2:default或 eks:my-cluster/default)。

要求格式

使用 HTTP POST 搭配下列標頭和內文來傳送日誌：

標頭：

- Authorization: Bearer <your-bearer-token>
- Content-Type: application/json

內文格式：

請求內文應為 JSON 格式，其中包含一系列事件：

```
{
    "event": [
        {
            "time": 1730141374.001,
            "event": "Application started successfully",
            "host": "web-server-1",
            "source": "application.log",
            "severity": "info"
        },
        {
            "time": 1730141374.457,
            "event": "User login successful",
            "host": "web-server-1",
            "source": "auth.log",
            "user": "john.doe"
        }
    ]
}
```

欄位描述：

- time – 以秒為單位的 Unix epoch 時間戳記，選用小數表示次秒精確度 （選用）
- event – 日誌訊息或事件資料 （必要）
- host – 來源主機名稱或識別符 （選用）
- source – 日誌來源識別符 （選用）

您可以視需要包含其他自訂欄位。

範例請求

```
curl -X POST "https://logs.<region>.amazonaws.com/services/collector/event?
logGroup=MyLogGroup&logStream=MyStream" \
  -H "Authorization: Bearer ACWL<token>" \
  -H "Content-Type: application/json" \
  -d '{"event":{"message":"User logged
 in","user_id":"u-123"},"time":1486683865.0,"host":"web-01","source":"auth-service"}'
```

最佳實務

批次處理事件

為了獲得更好的效能和效率：

- 盡可能在單一請求中批次處理多個事件
- 建議的批次大小：每個請求 10–100 個事件
- 請求大小上限：1 MB

錯誤處理

在應用程式中實作適當的錯誤處理。常見的 HTTP 狀態碼：

- 200 OK – 日誌已成功擷取
- 400 Bad Request – 無效的請求格式或參數
- 401 Unauthorized – 承載字符無效或過期
- 403 Forbidden – 許可不足
- 404 Not Found – 日誌群組或串流不存在

- `429 Too Many Requests` – 超過速率限制
- `500 Internal Server Error` – 服務錯誤 （以指數退避重試）

限制

- 事件大小上限：每個事件 256 KB
- 請求大小上限：1 MB
- 每個請求的事件上限：10，000
- 日誌群組名稱必須遵循 CloudWatch Logs 命名慣例
- 如果使用承載字符身分驗證，則必須在日誌群組上啟用承載字符身分驗證。

使用 NDJSON 端點傳送日誌 (ND-JSON 日誌）

ND-JSON Logs 端點 (`/ingest/bulk`) 接受 NDJSON （換行分隔 JSON) 格式的日誌。每一行只包含一個 JSON 值，以換行字元分隔。

如果您使用承載字符身分驗證，請先完成 中的設定步驟，the section called “承載字符身分驗證”再繼續。

要求格式

每行傳送一個 JSON 值，以 \n(LF) 或 \r\n(CRLF) 分隔。空白行會被無提示地忽略。

```
{"timestamp":1771007942000,"message":"event one","level":"INFO"}
{"timestamp":1771007943000,"message":"event two","level":"ERROR"}
{"timestamp":1771007944000,"message":"event three","level":"DEBUG"}
```

`application/json` 和 都`application/x-ndjson`被接受為 Content-Type。

接受的 JSON 值類型

根據 NDJSON 規格 (RFC 8259)，每行接受任何有效的 JSON 值。

JSON 物件 （最常見）：

```
{"timestamp":1771007942000,"message":"User logged in","service":"auth"}
{"timestamp":1771007943000,"error":"Connection timeout","service":"api"}
```

JSON 陣列（扁平化為個別事件）：

```
[{"timestamp":1000,"message":"a"},{"timestamp":2000,"message":"b"}]
```

此單行會產生 2 個事件。每個陣列元素都會成為個別的日誌事件。

基本值：

```
"a plain string log message"
42
true
null
```

每個基本會成為具有伺服器目前時間戳記的專屬事件。

混合類型：

```
{"timestamp":1771007942000,"message":"structured event"}
"unstructured string message"
42
{"timestamp":1771007943000,"error":"something failed"}
```

所有 4 行都被接受為有效事件。

行內容	Behavior (行為)
JSON 物件	已接受，如果有的話擷取時間戳記
JSON 陣列	平面化 – 每個元素都會成為個別的事件
空陣列 []	已接受，會產生 0 個事件
JSON 字串	接受為事件訊息
JSON 號碼	接受為事件訊息
JSON 布林值	接受為事件訊息
JSON null	接受為事件訊息

行內容	Behavior (行為)
無效的 JSON	已略過（已計算，處理會繼續）
空行	已忽略（未計為略過）

時間戳記欄位

"timestamp" 欄位以 epoch 毫秒（非秒）為單位。

格式	範例	解譯為
數值（毫秒）	"timestamp":1771007942000	1771007942000 毫秒
缺少	（無時間戳記欄位）	伺服器目前時間
非數值	"timestamp":"invalid"	伺服器目前時間
非物件行	"hello", 42, true	伺服器目前時間

無效的行

非有效 JSON 的行會無提示地略過並計數。處理會繼續進行下一行。

```
{"message":"valid event"}
this is not valid json
{"message":"another valid event"}
```

結果：擷取 2 個事件，略過 1 個事件。傳回 HTTP 200。

如果所有行都無效，會傳回 HTTP 400與 "All events were invalid"。

範例請求

```
curl -X POST "https://logs.<region>.amazonaws.com/ingest/bulk?
logGroup=MyLogGroup&logStream=MyStream" \
  -H "Authorization: Bearer ACWL<token>" \
```

```
  -H "Content-Type: application/x-ndjson" \
  -d '{"timestamp":1771007942000,"message":"User logged in","level":"INFO"}
{"timestamp":1771007943000,"message":"Query took 42ms","level":"DEBUG"}
{"timestamp":1771007944000,"error":"Connection refused","level":"ERROR"}'
```

回應

成功（接受所有事件）：

```
HTTP 200 OK
{}
```

部分成功（部分事件遭到拒絕）：

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1,
 \"expiredLogEventCount\": 1}"
  }
}
```

`rejectedLogRecords` 欄位是已拒絕事件的總數。`errorMessage` 欄位包含依拒絕原因分類的 JSON 編碼明細：

- `tooOldLogEventCount` – 時間戳記早於保留期間的事件
- `tooNewLogEventCount` – 未來時間戳記太久的事件
- `expiredLogEventCount` – 處理期間過期的事件

最佳實務

批次處理事件

為了獲得更好的效能和效率：

- 盡可能在單一請求中批次處理多個事件
- 建議的批次大小：每個請求 10–100 個事件
- 請求大小上限：1 MB

錯誤處理

在應用程式中實作適當的錯誤處理。常見的 HTTP 狀態碼：

- `200 OK` – 日誌已成功擷取
- `400 Bad Request` – 無效的請求格式或參數
- `401 Unauthorized` – 承載字符無效或過期
- `403 Forbidden` – 許可不足
- `404 Not Found` – 日誌群組或串流不存在
- `429 Too Many Requests` – 超過速率限制
- `500 Internal Server Error` – 服務錯誤 （以指數退避重試）

限制

- 事件大小上限：每個事件 256 KB
- 請求大小上限：1 MB
- 每個請求的事件上限：10，000
- 日誌群組名稱必須遵循 CloudWatch Logs 命名慣例
- 如果使用承載字符身分驗證，則必須在日誌群組上啟用承載字符身分驗證。

使用結構化 JSON 端點傳送日誌 （結構化 JSON 日誌）

結構化 JSON Logs 端點 (`/ingest/json`) 接受標準 JSON – 單一 JSON 物件或物件的 JSON 陣列。此端點專為結構式日誌資料而設計，其中每個事件都是 JSON 物件。

如果您使用承載字符身分驗證，請先完成 中的設定步驟，the section called “承載字符身分驗證”再繼續。

要求格式

僅`application/json`接受 做為 Content-Type。

單一 JSON 物件：

```
{"timestamp":1771007942000,"message":"single event","level":"INFO"}
```

物件的 JSON 陣列：

```
[
  {"timestamp":1771007942000,"message":"event one","level":"INFO"},
  {"timestamp":1771007943000,"message":"event two","level":"ERROR"}
]
```

接受的 JSON 值類型

此端點很嚴格 – 僅接受 JSON 物件做為事件。

Input	Behavior (行為)
單一 JSON 物件	接受為一個事件
物件的 JSON 陣列	每個物件都會成為個別的事件
空陣列 `[]`	已接受，會產生 0 個事件
陣列中的非物件 （ 字串、數字等 ）	略過
最上層基本 (`"hello"`、`42`)	略過
串連物件 `{...}{...}`	僅剖析第一個物件

範例 – 混合類型的陣列：

```
[
  {"timestamp":1771007942000,"message":"valid object"},
  "just a string",
  42,
  {"timestamp":1771007943000,"message":"another valid object"}
]
```

結果：擷取 2 個事件 （ 物件 ）、略過 2 個事件 （ 字串和數字 ）。

時間戳記欄位

`"timestamp"` 欄位以 epoch 毫秒為單位，與 NDJSON 端點相同。

格式	範例	解譯為
數值（毫秒）	`"timestamp":1771007942000`	1771007942000 毫秒
缺少	（無時間戳記欄位）	伺服器目前時間
非數值	`"timestamp":"invalid"`	伺服器目前時間

範例請求

```
curl -X POST "https://logs.<region>.amazonaws.com/ingest/json?
logGroup=MyLogGroup&logStream=MyStream" \
  -H "Authorization: Bearer ACWL<token>" \
  -H "Content-Type: application/json" \
  -d '[{"timestamp":1771007942000,"message":"User logged in","user_id":"u-123"},
{"timestamp":1771007943000,"message":"Order placed","order_id":"o-456"}]'
```

回應

成功（接受所有事件）：

```
HTTP 200 OK
{}
```

部分成功（部分事件遭到拒絕）：

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1,
 \"expiredLogEventCount\": 1}"
  }
}
```

`rejectedLogRecords` 欄位是已拒絕事件的總數。`errorMessage` 欄位包含依拒絕原因分類的 JSON 編碼明細：

- `tooOldLogEventCount` – 時間戳記早於保留期間的事件

- `tooNewLogEventCount` – 未來時間戳記太久的事件
- `expiredLogEventCount` – 處理期間過期的事件

最佳實務

批次處理事件

為了獲得更好的效能和效率：

- 盡可能在單一請求中批次處理多個事件
- 建議的批次大小：每個請求 10–100 個事件
- 請求大小上限：1 MB

錯誤處理

在應用程式中實作適當的錯誤處理。常見的 HTTP 狀態碼：

- `200 OK` – 日誌已成功擷取
- `400 Bad Request` – 無效的請求格式或參數
- `401 Unauthorized` – 承載字符無效或過期
- `403 Forbidden` – 許可不足
- `404 Not Found` – 日誌群組或串流不存在
- `429 Too Many Requests` – 超過速率限制
- `500 Internal Server Error` – 服務錯誤（以指數退避重試）

限制

- 事件大小上限：每個事件 256 KB
- 請求大小上限：1 MB
- 每個請求的事件上限：10，000
- 日誌群組名稱必須遵循 CloudWatch Logs 命名慣例
- 如果使用承載字符身分驗證，則必須在日誌群組上啟用承載字符身分驗證。

HTTP 擷取端點的比較

功能	HLC 日誌	ND-JSON 日誌	結構化 JSON 日誌	OpenTelemetry 日誌
路徑	`/services/collector/event`	`/ingest/bulk`	`/ingest/json`	`/v1/logs`
內容類型	`application/json`	`application/json` 或 `application/x-ndjson`	`application/json`	`application/json` 或 `application/x-protobuf`
時間戳記欄位	`"time"`（秒）	`"timestamp"`（毫秒）	`"timestamp"`（毫秒）	`"timeUnixNano"`（奈秒）
必要欄位	`"event"`	無	無	OTLP 結構 (`"resourceLogs"`)
部分成功回應	否	是	是	是
查詢參數支援	是	是	是	否（僅限標頭）
實體中繼資料	是	是	是	否
接受基本概念	否	是	否	否
行型剖析	否	是	否	否
Protobuf 支援	否	否	否	是
Retry-After 標頭	否	否	否	是

選擇端點

- 使用 HLC 格式？ 使用 HLC 日誌。您現有的 HLC 承載會以最少的變更運作。
- line-by-line串流日誌？ 使用 ND-JSON 日誌。最適合每行發出一個事件的日誌管道。最靈活 – 接受任何 JSON 值類型。
- 傳送結構化 JSON 承載？ 使用結構化 JSON 日誌。最適合產生格式正確的 JSON 物件或陣列的應用程式。
- 已使用 OpenTelemetry？ 使用 OpenTelemetry Logs。接受 OTLP JSON 或 Protobuf 格式，並支援部分成功回應與重試語意。

搭配 AWS SDK 使用 CloudWatch Logs

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

如需 CloudWatch Logs 的範例，請參閱 使用 SDK 的 CloudWatch Logs 程式碼範例 AWS SDKs。

可用性範例

找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

日誌管理

CloudWatch Logs 提供進階日誌管理功能，可協助您更有效地組織、轉換和分析日誌資料。這些功能包括跨帳戶和跨區域資料集中化、自動資料探索和結構描述管理、擷取期間的日誌轉換，以及使用面向進行互動式日誌探索的增強型分析。

主題

- 資料來源探索和管理
- 資料來源啟用的功能
- AWS 服務 支援資料來源
- 資料來源支援的第三方來源

資料來源探索和管理

CloudWatch Logs 會根據資料來源和類型自動探索和分類您的日誌資料，讓您更輕鬆地大規模了解和管理日誌。此功能為 AWS Amazon VPC Flow Logs、CloudTrail 和 Route 53 等付費來源，以及第三方安全工具提供結構描述探索。

Logs Management 主控台提供依資料來源和類型整理的日誌的高階檢視，而不只是日誌群組。此組織可協助您：

- 檢視依 AWS 服務、第三方來源 （ 例如 Okta 或 CrowdStrike) 和自訂來源分類的日誌
- 自動了解日誌資料的結構描述和結構
- 根據探索到的結構描述欄位建立欄位索引政策
- 更有效率地管理不同資料來源的日誌
- 依不同資料來源查詢日誌

當您為支援 AWS 的服務啟用 CloudWatch Logs 記錄時，CloudWatch Logs 會自動將適當的結構描述套用至您的日誌。此自動結構描述應用程式有助於維持一致性，並提供日誌結構的即時洞見。

什麼是 CloudWatch Logs 資料來源？

CloudWatch Logs 資料來源是一項功能，提供根據產生日誌的來源組織和分類日誌資料的新方法。雖然 CloudWatch Logs 傳統上會使用日誌群組來組織日誌，但資料來源提供額外的組織層，依其原始服務和日誌類型將日誌分組。

資料來源的運作方式

資料來源提供以服務為基礎的日誌組織，並簡化整個 AWS 基礎設施的探索。您可以輕鬆從特定服務找到日誌，並依日誌類型進行篩選，而不需要知道個別日誌群組名稱或結構。

對於第三方來源和選用的應用程式日誌來源，資料來源使用 CloudWatch 管道來分類您的日誌。當您設定管道以擷取和轉換日誌時，您可以指定資料來源名稱和類型。然後CloudWatch Logs 會自動分類管道處理的所有日誌。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 CloudWatch 管道。 Amazon CloudWatch

資料來源使用兩個金鑰識別符來分類日誌：

- 資料來源名稱：產生日誌 AWS 的服務、第三方來源或應用程式 （ 例如 Route 53、Amazon VPC、CloudTrail、Okta SSO 或 CrowdStrike Falcon)。
- 資料來源類型：該服務產生的特定日誌類型。

結構描述會定義日誌資料的結構，包括有哪些欄位存在，以及如何組織資訊。單一資料來源可以產生具有不同結構描述和用途的多種日誌類型。例如， AWS CloudTrail 資料來源有兩種類型：管理事件 （ 追蹤控制平面操作，例如建立或刪除資源 ） 和資料事件 （ 追蹤資料平面操作，例如 S3 物件存取 ）。每種類型都有不同的結構描述，因為它們會擷取不同類型的資訊。

如何開始

CloudWatch Logs 會根據日誌的原始伺服器，將日誌分類為資料來源。方法取決於您正在使用的日誌類型：

AWS 服務 日誌

來自支援的 AWS 服務日誌會依資料來源自動分組，而不需要任何組態。CloudWatch Logs 會辨識這些日誌，並根據原始服務套用適當的資料來源名稱和類型。

第三方日誌

第三方日誌需要用於資料來源分類的管道。當您設定管道從支援的第三方來源擷取日誌時，例如 Microsoft Office 365、Okta、CrowdStrike 或 Palo Alto Networks，您可以在管道組態中指定資料來源名稱和類型。CloudWatch Logs 會自動分類管道使用這些識別符處理的所有日誌。

管道可以選擇性地將第三方日誌轉換為開放網路安全結構描述架構 (OCSF) 格式，以進行標準化安全事件分析。啟用 OCSF 轉換時，資料來源名稱和類型會根據 OCSF 結構描述映射自動決定。如果沒有 OCSF 轉換，您可以指定資料來源名稱，並在管道組態中輸入 。

應用程式記錄

對於自訂應用程式日誌，您可以使用下列其中一種方法來依資料來源進行分類：

- 日誌群組標籤 - 使用 金鑰將標籤新增至日誌群組`cw:datasource:type`，`cw:datasource:name`並分別指定日誌群組中擷取之所有日誌的資料來源名稱和類型。標籤值最多可達 64 個字元，且只能包含小寫字母、數字和底線。它們必須以字母或數字開頭，而且不能包含雙底線 (__)。
- 管道組態 - 在擷取應用程式日誌時，透過日誌處理管道設定資料來源資訊。

> **Note**
>
> 資料來源名稱不能以「aws」或「amazon」開頭，以避免 AWS 與服務日誌衝突。

系統欄位

CloudWatch Logs 會自動將三個系統欄位新增至依資料來源分類的日誌。這些欄位做為預設面向：

- `@data_source_name` - 包含資料來源的名稱，如果未確定則為「未知」
- `@data_source_type` - 包含資料來源的類型，如果未確定則為「未知」
- `@data_format` - 指出日誌資料的格式

當無法判斷資料來源名稱或類型時，這些欄位會設定為「未知」。在主控台的「日誌管理」下，具有「未知」值的資料來源仍然可見於構面和資料來源資料表中，可讓您識別未分類的日誌及其來源的日誌群組。

`@data_format` 欄位可包含下列其中一個值：

- `Default` - 在不修改的情況下擷取的日誌。
- `Custom` - 透過管道處理器處理的日誌，或擷取至具有資料來源名稱/類型標籤的日誌群組的日誌。
- `OCSF-<version>` - 在管道中使用 OCSF （開放式網路安全結構描述架構） 處理器處理的日誌。
- `AWS-OTEL-LOG-V<version>` - 透過 CloudWatch OTLP 端點擷取的 OpenTelemetry 日誌。
- `AWS-OTEL-TRACE-V<version>` - 透過 CloudWatch OTLP 端點擷取的 OpenTelemetry 追蹤。

這些系統欄位可讓您根據日誌的來源和格式篩選和查詢日誌，讓您更輕鬆地使用來自不同原始伺服器和處理管道的日誌。

存取資料來源

主控台

在 CloudWatch Logs 主控台中，您可以使用日誌管理索引標籤來存取資料來源。CloudWatch Logs 會根據資料來源和類型自動合併您的日誌資料，持續探索新擷取的資料。從資料來源清單中，您可以建立管道、定義欄位索引和面向。

AWS CLI

使用下列命令列出帳戶中不同的資料來源和日誌類型：

```
aws logs list-aggregate-log-group-summaries --group-by DATA_SOURCE_NAME_AND_TYPE
```

與日誌群組的關係

資料來源補充而不是取代日誌群組。您的日誌會繼續像以前一樣存放在日誌群組中，但現在它們也會自動標記資料來源資訊。此雙組織可讓您：

- 使用日誌群組進行精細存取控制和保留政策
- 使用資料來源進行服務型日誌探索和分析
- 根據您的需求，使用任一組織方法查詢日誌

資料來源可讓您更輕鬆地大規模使用日誌，方法是在整個 AWS 基礎設施中提供以服務為中心的日誌資料檢視。

資料來源啟用的功能

資料來源可透過欄位探索和一致的資料結構來啟用進階日誌處理和分析功能。

- 面向：面向是索引日誌欄位，提供互動式篩選和分析，無需撰寫查詢。CloudWatch Logs 會自動建立資料來源名稱和類型的面向，您可以在探索到的欄位上建立面向政策，以加速故障診斷。面向會在 CloudWatch Logs Insights 中顯示值分佈和計數，以便透過point-and-click探索輕鬆識別模式。
- 管道：建立適用於特定資料來源名稱和類型之所有日誌的轉換管道。這可讓您為來自相同來源的日誌定義一致的處理規則。

- 欄位探索：CloudWatch Logs 會根據管道處理器，自動探索每個資料來源名稱和類型組合的欄位及其資料類型。對於 AWS 受管日誌，會預先定義欄位結構。對於應用程式日誌，我們建議您維持一致的日誌格式，以最大限度地提高與分析工具的相容性，例如需要明確定義欄位結構的 Amazon S3 資料表。

您可以使用 `GetLogFields` API 檢視任何資料來源的欄位及其類型的完整清單：

```
aws logs get-log-fields --data-source-name <name>  --data-source-type <type>
```

此欄位探索和一致性可啟用進階分析和整合，因為外部工具在處理日誌資料時可以使用可預測的欄位結構。

AWS 服務 支援資料來源

下表列出由 CloudWatch Logs AWS 服務 自動分類為資料來源的 ：

資料來源名稱 (@data_source_name 欄位)	資料來源類型 (@data_source_type 欄位)
amazon_api_gateway	access
amazon_bedrock_agentcore	browser_usage
amazon_bedrock_agentcore	code_interpreter_application
amazon_bedrock_agentcore	code_interpreter_usage
amazon_bedrock_agentcore	gateway_application
amazon_bedrock_agentcore	identity_workload_application
amazon_bedrock_agentcore	memory_application
amazon_bedrock_agentcore	online_evaluation_config
amazon_bedrock_agentcore	runtime_application
amazon_bedrock_agentcore	runtime_usage
amazon_bedrock_agents	application

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
amazon_bedrock_agents	event
amazon_bedrock_knowledge_bases	application
amazon_cloudfront	access
amazon_cloudfront	connection
amazon_cloudwatch	rum_app_monitor
amazon_cognito	user_pool
amazon_ec2	verified_access
amazon_eks	api_server
amazon_eks	audit
amazon_eks	authenticator
amazon_eks	controller_manager
amazon_eks	scheduler
amazon_elasticache	cluster
amazon_eventbridge	eventbus_error
amazon_eventbridge	eventbus_info
amazon_eventbridge	pipes_execution
amazon_interactive_video_service	chat
amazon_managed_prometheus	scraper
amazon_managed_prometheus	workspace
amazon_msk	broker

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
amazon_msk	connect
amazon_opensearch_service	pipeline
amazon_q_business	events
amazon_q_business	sync_job
amazon_q_connect	events
amazon_route53	global_resolver_query
amazon_route53	hosted_zones
amazon_route53	profiles_resolver_query
amazon_route53	resolver_query
amazon_sagemaker	workteam_activity
amazon_ses	ingress_endpoints
amazon_ses	rule_sets
amazon_ses	traffic_policy
amazon_vpc	flow
amazon_vpc	route_server_peer
amazon_vpc_lattice	access
amazon_vpc_lattice	resource_access
amazon_workmail	access_control
amazon_workmail	authentication
amazon_workmail	personal_access

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
amazon_workmail	workmail_access
amazon_workmail	workmail_availability
aws_b2b_data_interchange	execution
aws_backup	data_access
aws_backup	hypervisor
aws_clean_rooms	analysis
aws_client_vpn	connection
aws_client_vpn	event
aws_cloudtrail	data
aws_cloudtrail	management
aws_elemental_mediapackage	egress_access
aws_elemental_mediapackage	ingress_access
aws_elemental_mediatailor	ad_decision
aws_elemental_mediatailor	manifest
aws_elemental_mediatailor	transcode
aws_entity_resolution	id_mapping_workflow
aws_entity_resolution	matching_workflow
aws_iot_fleetwise	error
aws_mainframe_modernization	batch_job
aws_mainframe_modernization	config

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
aws_mainframe_modernization	console
aws_mainframe_modernization	dataset_import
aws_network_firewall	alert
aws_network_firewall	flow
aws_network_firewall	tls
aws_nlb	access
aws_pcs	job_completion
aws_pcs	scheduler
aws_security_hub_cspm	asff_finding
aws_shield	protection_flow
aws_step_functions	express
aws_step_functions	standard
aws_transfer_family	server
aws_waf	access

資料來源支援的第三方來源

下表列出透過管道擷取時，CloudWatch Logs 自動分類為資料來源的第三方來源：

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
crowdstrike_falcon	detection_finding
crowdstrike_falcon	process_activity

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
github_auditlogs	account_change
github_auditlogs	api_activity
github_auditlogs	entity_management
microsoft_entraid	account_change
microsoft_entraid	authentication
microsoft_entraid	entity_management
microsoft_entraid	user_access_management
microsoft_office365	account_change
microsoft_office365	application_lifecycle
microsoft_office365	authentication
microsoft_office365	compliance_finding
microsoft_office365	detection_finding
microsoft_office365	email_activity
microsoft_office365	file_hosting_activity
microsoft_office365	group_management
microsoft_office365	incident_finding
microsoft_office365	user_access_management
microsoft_office365	vulnerability_finding
microsoft_office365	web_resources_activity
microsoft_windows	account_change

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
microsoft_windows	authentication
microsoft_windows	entity_management
microsoft_windows	event_log_activity
microsoft_windows	file_system_activity
microsoft_windows	group_management
microsoft_windows	kernel_activity
okta_auth0	api_activity
okta_auth0	authentication
okta_sso	api_activity
okta_sso	authentication
okta_sso	detection_finding
okta_sso	entity_management
paloaltonetworks_nextgenera tionfirewall	authentication
paloaltonetworks_nextgenera tionfirewall	detection_finding
paloaltonetworks_nextgenera tionfirewall	network_activity
paloaltonetworks_nextgenera tionfirewall	process_activity
sentinelone_endpointsecurity	dns_activity
sentinelone_endpointsecurity	file_system_activity

資料來源名稱 (@data_source_name 欄位）	資料來源類型 (@data_source_type 欄位）
sentinelone_endpointsecurity	http_activity
sentinelone_endpointsecurity	process_activity
servicenow_cmdb	api_activity
servicenow_cmdb	datastore_activity
servicenow_cmdb	entity_management
wiz_cnapp	api_activity
wiz_cnapp	authentication
wiz_cnapp	compliance_finding
wiz_cnapp	detection_finding
wiz_cnapp	vulnerability_finding
zscaler_internetaccess	authentication
zscaler_internetaccess	dns_activity
zscaler_internetaccess	http_activity
zscaler_internetaccess	network_activity

使用 CloudWatch Logs Insights 分析日誌資料

使用 CloudWatch Logs Insights，您可以在 Amazon CloudWatch Logs 中以互動方式搜尋和分析日誌資料。您可以執行查詢，協助您更有效率且有效地回應操作問題。除了使用日誌群組查詢之外，您還可以使用面向、資料來源和資料類型進行查詢。如果發生問題，您可以使用 CloudWatch Logs 來識別可能的原因並驗證已部署的修正。每個帳戶僅限 100 個並行 CloudWatch Logs Insights QL，包括新增至儀表板的查詢。此外，您可以為 OpenSearch Service PPL 或 OpenSearch Service SQL 執行 15 個並行查詢。

CloudWatch Logs Insights 支援三種查詢語言，可用於查詢：

- 專用 Logs Insights 查詢語言 (Logs Insights QL)，具有幾個簡單但強大的命令。
- OpenSearch Service 管道處理語言 (PPL)。OpenSearch PPL 可讓您使用一組以管道 (|) 分隔的命令來分析日誌。

  使用 OpenSearch PPL，您可以使用一起輸送的命令來擷取、查詢和分析資料，讓您更輕鬆地了解和編寫複雜的查詢。語法可讓命令鏈結轉換和處理資料。使用 PPL，您可以篩選和彙總資料，並使用一組豐富的數學、字串、日期、條件式和其他函數進行分析。
- OpenSearch Service 結構化查詢語言 (SQL)。使用 OpenSearch SQL 查詢，您可以宣告方式分析日誌。您可以使用 SELECT、 FROM、WHERE、GROUP BY、HAVING 等命令，以及 SQL 中可用的各種其他命令和函數。您可以跨日誌群組執行 JOINs、使用子查詢關聯日誌之間的資料，並使用一組豐富的 JSON、數學、字串、條件式和其他 SQL 函數對日誌執行強大的分析。

  當您使用 SQL 或 PPL 命令時，請務必在反引號中以特殊字元（非字母和非數字）括住欄位，以成功查詢它們。例如，將 `@message`、 `Operation.Export`和 括在反引號`Test::Field`中。您不需要在反引號中以純字母名稱括住欄位。

CloudWatch Logs Insights 提供下列功能，可與任何查詢語言搭配使用。

- 從 Amazon Route 53 AWS Lambda、 和 Amazon VPC 等服務 AWS CloudTrail自動探索日誌中的日誌欄位，以及以 JSON 形式發出日誌事件的任何應用程式或自訂日誌。 AWS
- 建立欄位索引以降低成本和加速結果，尤其是針對大量日誌群組或日誌事件的查詢。在建立日誌事件中常見欄位的欄位索引之後，您可以在查詢中使用它們。查詢會略過處理已知不包含索引欄位的日誌事件，並處理較少的資料。

> **Note**
>
> `filterIndex` 命令僅適用於 Logs Insights QL。

- 偵測和分析日誌事件中的模式。模式是在您的日誌欄位之間重複出現的共用文字結構。當您檢視查詢的結果時，您可以選擇模式索引標籤，以查看 CloudWatch Logs 根據結果範例找到的模式。
- 儲存查詢、查看您的查詢歷史記錄、重新執行已儲存的查詢，以及搭配參數使用已儲存的查詢。
- 將查詢新增至儀表板。
- 使用 加密查詢結果 AWS Key Management Service。
- 使用自然語言產生查詢可讓您使用自然語言來建立 CloudWatch Logs Insights 查詢。您可以詢問問題或描述您要尋找的資料，然後 AI 會根據您的提示產生查詢，並提供查詢運作方式的line-by-line說明。
- 使用面向來分組、篩選和互動探索您的日誌。

只有在您使用 Logs Insights QL 時，才支援下列 CloudWatch Logs Insights 功能。

- 比較查詢，將日誌群組中的日誌事件與上一個時段的日誌事件進行比較。

> **Important**
>
> CloudWatch Logs Insights 無法存取時間戳記早於日誌群組建立時間的日誌事件。

如果您登入的帳戶設定為 CloudWatch 跨帳戶觀察功能中的監控帳戶，則可以對連結至此監控帳戶之來源帳戶中的日誌群組執行 CloudWatch Logs Insights 查詢。您可以查詢位於不同帳戶中的多個日誌群組。如需詳細資訊，請參閱 CloudWatch 跨帳戶觀察功能。

當您使用 Logs Insights QL 建立查詢時，您也可以使用自然語言來建立 CloudWatch Logs Insights 查詢。因此，請提出問題或描述您正在尋找的資料。此 AI 輔助功能會根據您的提示產生查詢，並逐行說明查詢的運作方式。如需詳細資訊，請參閱使用自然語言來產生和更新 CloudWatch Logs Insights 查詢。

使用任何支援的查詢語言的查詢，如果尚未完成，會在 60 分鐘後逾時。查詢結果提供七天。

無論查詢語言為何，CloudWatch Logs Insights 查詢都會根據查詢的資料量產生費用。如需詳細資訊，請參閱 Amazon CloudWatch 定價。

您可以使用 CloudWatch Logs Insights 來搜尋在 2018 年 11 月 5 日或之後傳送至 CloudWatch Logs 的日誌資料。

> **Important**
>
> 如果您的網路安全團隊不允許使用 web 通訊端，則您目前無法存取 CloudWatch 主控台的 CloudWatch Logs Insights 部分。您可以透過 API 來使用 CloudWatch Logs Insights 查詢功能。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 StartQuery。

目錄

支援的查詢語言

下列各節列出每種查詢語言支援的命令。它們也會描述語法格式並提供範例查詢。

主題

CloudWatch Logs Insights 查詢語言 (Logs Insights QL)

本節包含 Logs Insights QL 命令和函數的完整文件。它也包含此語言的範例查詢。

如需有關您可以使用的其他查詢語言的資訊，請參閱 OpenSearch Service PPL、OpenSearch Service SQL 和 CloudWatch。 Metrics Insights

主題

CloudWatch Logs Insights 語言查詢語法

本節提供有關 Logs Insights QL 的詳細資訊。查詢語法支援不同的函式和運算，包含但不限於一般函式、算術和比較運算，以及正規表達式。

> **Important**
>
> 為了避免因執行大型查詢而產生過多費用，請記住下列最佳實務：
>
> - 為每個查詢僅選取必要的日誌群組。
> - 一律為您的查詢指定最窄的可能時間範圍。
> - 當您使用主控台執行查詢時，請先取消所有查詢，再關閉 CloudWatch Logs Insights 主控台頁面。否則，查詢會繼續執行直到完成。
> - 當您將 CloudWatch Logs Insights 小工具新增至儀表板時，請確定儀表板不會以高頻率重新整理，因為每次重新整理都會啟動新的查詢。

若要建立包含多個命令的查詢，請使用直立線符號字元 (|) 分隔命令。

若要建立包含註解的查詢，請使用雜湊字元 (#) 作為註解的開頭。

> **Note**
>
> CloudWatch Logs Insights 會自動探索不同日誌類型的欄位，並產生以 @ 字元開頭的欄位。如需進一步了解這些自動產生的欄位，請參閱《Amazon CloudWatch 使用者指南》中的支援的日誌和探索的欄位。

下表簡要描述每個命令。此資料表下面是每個命令的詳細說明，並附有範例。

> **Note**
>
> 標準日誌類別中的日誌群組支援所有 Logs Insights QL 查詢命令。不常存取日誌類別中的日誌群組支援所有 Logs Insights QL 查詢命令`pattern`，但 `diff`、 和 除外`unmask`。

anomaly	使用機器學習識別日誌資料中的異常模式。
display	在查詢結果中顯示一個或多個特定欄位。
fields	在查詢結果中顯示特定欄位，並支援可用於修改欄位值和建立要在查詢中使用之新欄位的函數和操作。
filter	篩選查詢以僅傳回符合一個或多個條件的日誌事件。
filterIndex	強制查詢只嘗試掃描在欄位索引中提到的欄位上編製索引的日誌群組，並同時包含該欄位索引的值。這透過嘗試僅掃描這些日誌群組中包含此欄位索引查詢中指定值的日誌事件來減少掃描的磁碟區。 不常存取日誌類別中的日誌群組不支援此命令。
pattern	自動將您的日誌資料叢集化，以形成模式。模式是指日誌欄位之間反覆出現的共同文字結構。CloudWatch Logs Insights 可讓您分析日誌事件中找到的模式。如需詳細資訊，請參閱模式分析。
diff	將請求時段中找到的日誌事件與先前相同長度時段的日誌事件進行比較，以便您可以尋找趨勢並了解特定日誌事件是否為新事件。
parse	從日誌欄位擷取資料，建立一個您可在查詢中處理的擷取欄位。**parse** 支援使用萬用字元的 glob 模式和規則運算式。

sort	以遞增 (asc) 或遞減 (desc) 方式顯示傳回的日誌事件。
SOURCE	在查詢SOURCE中包含 是根據要在查詢中包含的日誌群組名稱字首、帳戶識別符和日誌群組類別來指定大量日誌群組的有用方法。只有在您在中 AWS CLI 或以程式設計方式建立查詢時，才支援此命令，而不是在 CloudWatch 主控台中。
stats	使用日誌欄位值計算彙總統計數字。
limit	指定您希望查詢傳回的日誌事件數目上限。使用 **sort** 可傳回「前 20 個」或「最近 20 個」結果。
dedup	根據您指定之欄位中的特定值移除重複的結果。
unmask	顯示因為資料保護政策而遮罩某些內容的某個日誌事件的全部內容。如需有關日誌群組中資料保護的詳細資訊，請參閱 使用遮罩功能協助保護敏感日誌資料。
unnest	扁平化做為輸入的清單，以針對清單中的每個元素產生具有單一記錄的多個記錄。
lookup	透過比對欄位值，使用查詢資料表中的資料來豐富日誌事件。使用查詢資料表將參考資料，例如使用者詳細資訊、應用程式名稱或產品資訊新增至查詢結果。
其他操作和函數	CloudWatch Logs Insights 也支援許多比較、算術、日期時間、數值、字串、IP 地址以及一般函數和操作。

以下各節提供有關 CloudWatch Logs Insights 查詢命令的詳細資訊。

主題

- 日誌類別中支援的 Logs Insights QL 命令
- 異常
- display
- fields
- 篩選條件
- filterIndex

- SOURCE
- pattern
- 差異
- parse
- sort
- 統計資料
- limit
- dedup
- unmask
- 解巢狀
- 查詢
- 布林值、比較、數值、日期時間和其他函數
- 包含特殊字元的欄位
- 在查詢中使用別名和註解

日誌類別中支援的 Logs Insights QL 命令

標準日誌類別中的日誌群組支援所有 Logs Insights QL 查詢命令。不常存取日誌類別中的日誌群組支援 `pattern`、`filterIndex`、 `diff`和 以外的所有查詢命令`unmask`。

異常

使用 `anomaly`以機器學習自動識別日誌資料中的異常模式和潛在問題。

`anomaly` 命令會擴展現有`pattern`功能，並利用進階分析來協助識別日誌資料中的潛在異常。您可以使用 `anomaly`來減少在日誌中自動浮現異常模式或行為來識別和解決操作問題所需的時間。

`anomaly` 命令會搭配 `pattern`命令使用，先識別日誌模式，然後偵測這些模式中的異常。您也可以`anomaly`結合 `filter`或 `sort`命令，將異常偵測聚焦於資料的特定子集。

異常命令輸入

`anomaly` 命令通常會在 `pattern`命令之後使用，以分析日誌資料中識別的模式。命令不需要其他參數，並分析查詢中先前命令的輸出。

已識別的異常類型

`anomaly` 命令會識別五種不同的異常類型：

- 模式頻率異常：特定日誌模式的異常頻率，例如應用程式開始產生比平常更多的錯誤訊息時。
- 新模式異常：先前看不到的日誌模式可能表示日誌中出現新類型的錯誤或訊息。
- 字符變化異常：日誌訊息內容的非預期變更，可能表示預期日誌格式的異常變化。
- 數值符記異常：日誌中數值的異常變更，可協助偵測潛在的效能問題或非預期的指標變化。
- HTTP 錯誤碼異常：與 HTTP 錯誤回應相關的模式，在監控 Web 應用程式和 APIs時特別有用。

異常命令輸出

anomaly 命令會保留輸入資料中的所有欄位，並新增異常偵測結果，以協助識別日誌資料中的異常模式。

範例

下列命令會識別日誌資料中的模式，然後偵測這些模式中的異常：

```
fields @timestamp, @message
| pattern @message
| anomaly
```

anomaly 命令可與篩選搭配使用，以專注於特定日誌類型：

```
fields @timestamp, @message
| filter @type = "REPORT"
| pattern @message
| anomaly
```

anomaly 命令可與排序結合，以組織結果：

```
fields @timestamp, @message
| filter @type = "ERROR"
| pattern @message
| anomaly
| sort @timestamp desc
```

display

使用 display 在查詢結果中顯示一個或多個特定欄位。

display 命令僅顯示您指定的欄位。如果您的查詢包含多個 display 命令，查詢結果僅會顯示您在最終 display 命令中指定的欄位。

範例：顯示一個欄位

程式碼片段會顯示一個查詢範例，其使用剖析命令從 @message 中擷取資料，建立擷取欄位 loggingType 和 loggingMessage。查詢會傳回 loggingType 的值為 ERROR 的所有日誌事件。display 僅在查詢結果中顯示 loggingMessage 的值。

```
fields @message
| parse @message "[*] *" as loggingType, loggingMessage
| filter loggingType = "ERROR"
| display loggingMessage
```

> **Tip**
>
> 在查詢中僅使用一次 display。如果您在查詢中多次使用 display，則查詢結果只會顯示您最後一次使用 display 命令時指定的欄位。

fields

使用 fields 在查詢結果中顯示特定欄位。

如果您的查詢包含多個 fields 命令且未包含 display 命令，則結果會顯示在 fields 命令中指定的所有欄位。

範例：顯示特定欄位

下列範例顯示一個查詢，它傳回 20 個日誌事件並按降序顯示它們。會在查詢結果中顯示 @timestamp 和 @message 的值。

```
fields @timestamp, @message
| sort @timestamp desc
| limit 20
```

當您想要使用 支援的不同函數和操作fields來修改欄位值，並建立新的欄位以用於查詢display時，請使用 fields而非 。

您可以搭配使用 fields 命令和關鍵字 as，在日誌事件中建立使用欄位和函數的擷取欄位。例如：fields ispresent as isRes 會建立一個名為 isRes 的擷取欄位，而擷取欄位可在其餘查詢中使用。

篩選條件

使用 `filter` 來取得與一個或多個條件相符的日誌事件。

範例：使用一個條件篩選日誌事件

程式碼片段會顯示一個查詢範例，其會傳回 `range` 的值大於 3000 的所有日誌事件。該查詢將結果限制為 20 筆日誌事件，並按照 `@timestamp` 依遞減順序對日誌事件進行排序。

```
fields @timestamp, @message
| filter (range>3000)
| sort @timestamp desc
| limit 20
```

範例：使用多個條件篩選日誌事件

您可以使用關鍵字 `and` 和 `or` 以結合多個條件。

程式碼片段會顯示一個查詢範例，其會傳回 `range` 的值大於 3000 且 `accountId` 的值等於 123456789012 的日誌事件。該查詢將結果限制為 20 筆日誌事件，並按照 `@timestamp` 依遞減順序對日誌事件進行排序。

```
fields @timestamp, @message
| filter (range>3000 and accountId=123456789012)
| sort @timestamp desc
| limit 20
```

索引欄位和 filter 命令

如果您已為日誌群組建立欄位索引，您可以利用這些欄位索引，讓您的`filter`查詢更有效率並減少掃描的磁碟區。例如，假設您已建立 的欄位索引`requestId`。然後，該日誌群組上包含`filter requestId = value`或`filter requestId IN [value, value, ...]`將嘗試略過處理已知不包含索引欄位的日誌事件的任何 CloudWatch Logs Insights 查詢。透過嘗試僅掃描已知包含該索引欄位的日誌事件，可以減少掃描磁碟區，並且查詢更快。

如需欄位索引以及如何建立它們的詳細資訊，請參閱 建立欄位索引以改善查詢效能並減少掃描磁碟區。

> **Important**
>
> 只有具有 `filter` *`fieldName`* `=...`和 的查詢`filter` *`fieldName`* `IN...`將受益於欄位索引改進。使用 的查詢`filter` *`fieldName`* `like`不使用索引，並一律掃描所選日誌群組中的所有日誌事件。

範例：使用索引尋找與特定請求 ID 相關的日誌事件

此範例假設您已在 上建立欄位索引`requestId`。對於使用此欄位索引的日誌群組，查詢會利用欄位索引嘗試掃描最少數量的日誌事件，以尋找值`requestId`為 的事件 123456

```
fields @timestamp, @message
| filter requestId = "1234656"
| limit 20
```

filter 命令中的比對和規則表達式

篩選命令支援使用規則表達式。您可以使用下列比較運算子 (=、!=、<、<=、>、>=) 和布林值運算子 (and、or 以及 not)。

您可以使用關鍵字 in 來測試設定的成員資格並檢查陣列中的元素。若要檢查陣列中的元素，將陣列放在 in 之後。您可以搭配 in 使用布林運算子 not。您可以建立查詢來使用 in 傳回欄位為字串相符的日誌事件。欄位必須是完整的字串。例如，下列程式碼片段會顯示查詢使用 in 來傳回欄位 logGroup 是完整的字串 example_group 的日誌事件。

```
fields @timestamp, @message
| filter logGroup in ["example_group"]
```

您可以使用關鍵字詞 like 和 not like 來比對子字串。您可以使用規則表達式運算子 =~ 來比對子字串。若要比對帶有 like 和 not like 的子字串，請將要比對的子字串放在單引號或雙引號中。您可以搭配 like 和 not like 使用規則表達式模式。若要使用規則表達式運算子來比對子字串，請以斜線括住想要比對的子字串。下列範例包含程式碼片段，示範如何使用 filter 命令來比對子字串。

範例：比對子字串

以下範例會傳回 f1 含有單字 Exception 的日誌事件。所有三個範例都會區分大小寫。

第一個範例比對帶有 like 的子字符。

```
fields f1, f2, f3
| filter f1 like "Exception"
```

第二個範例比對帶有 like 和規則表達式模式的子字串。

```
fields f1, f2, f3
| filter f1 like /Exception/
```

第三個範例會比對子字串與規則表達式。

```
fields f1, f2, f3
| filter f1 =~ /Exception/
```

範例：比對子字串與萬用字元

您可以使用句點符號 (.) 作為規則表達式中的萬用字元來比對子字串。在下列範例中，查詢會傳回與以字串 ServiceLog 開始的 f1 的值相符項目。

```
fields f1, f2, f3
| filter f1 like /ServiceLog./
```

您可以在句點符號 (.*) 後面放置一個星號符號，來建立窮盡數量詞，窮盡數量詞會傳回儘可能多的相符項目。例如，以下查詢會傳回與以字串 ServiceLog 開始而且還包含字串 ServiceLog 的 f1 的值相符項目。

```
fields f1, f2, f3
| filter f1 like /ServiceLog.*/
```

可能的相符項目格式如下所示：

- ServiceLogSampleApiLogGroup
- SampleApiLogGroupServiceLog

範例：從相符項目中排除子字串

以下範例會顯示會傳回日誌事件的查詢，傳回的日誌事件中 f1 不會含有單字 Exception。這個範例區分大小寫。

```
fields f1, f2, f3
| filter f1 not like "Exception"
```

範例：比對區分大小寫的子字串

您可以比對帶有 `like` 和規則表達式且區分大小寫的子字串。請將下列參數 (?i) 放置在想要比對的子字串之前。下列範例會顯示會傳回日誌事件的查詢，傳回的日誌事件中 `f1` 會含有單字 Exception 或 exception。

```
fields f1, f2, f3
| filter f1 like /(?i)Exception/
```

filterIndex

`filterIndex` 使用 僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。對於在此欄位上編製索引的這些日誌群組，它會略過沒有包含索引欄位查詢中指定欄位的任何日誌事件的日誌群組，以進一步最佳化查詢。它透過嘗試僅掃描這些日誌群組中符合此欄位索引查詢中指定值的日誌事件，進一步減少掃描的磁碟區。如需欄位索引以及如何建立它們的詳細資訊，請參閱[建立欄位索引以改善查詢效能並減少掃描磁碟區](#)。

透過將實際搜尋空間限制為具有欄位索引的日誌群組和日誌事件，使用 `filterIndex`搭配索引欄位可協助您有效率地查詢包含 PB 日誌資料的日誌群組。

例如，假設您已在帳戶中的某些日誌群組`IPaddress`中為 建立欄位索引。然後，您可以建立下列查詢，並選擇查詢帳戶中的所有日誌群組，以尋找包含 `198.51.100.0` `IPaddress` 欄位中值的日誌事件。

```
fields @timestamp, @message
| filterIndex IPaddress = "198.51.100.0"
| limit 20
```

`filterIndex` 命令會導致此查詢嘗試略過未為 編製索引的所有日誌群組`IPaddress`。此外，在編製索引的日誌群組中，查詢會略過具有 `IPaddress` 欄位，但未觀察`198.51.100.0`為該欄位值的日誌事件。

使用 `IN`運算子將結果擴展到索引欄位的多個值中的任何一個。下列範例會尋找 `198.51.100.1`欄位中包含 值`198.51.100.0`或 的日誌事件`IPaddress`。

```
fields @timestamp, @message
```

```
| filterIndex IPaddress in ["198.51.100.0", "198.51.100.1"]
| limit 20
```

CloudWatch Logs 為標準日誌類別中的所有日誌群組提供預設欄位索引。預設欄位索引會自動用於下列欄位：

- @logStream
- @aws.region
- @aws.account
- @source.log
- @data_source_name
- @data_source_type
- @data_format
- traceId
- severityText
- attributes.session.id

CloudWatch Logs 也提供特定資料來源名稱和類型組合的預設欄位索引。預設欄位索引會自動用於下列資料來源名稱和類型組合：

資料來源名稱和類型	預設欄位索引
amazon_vpc.flow	action logStatus region flowDirection type
amazon_route53.resolver_query	query_type transport rcode

資料來源名稱和類型	預設欄位索引
aws_waf.access	action httpRequest.country
aws_cloudtrail.data aws_cloudtrail.management	eventSource eventName awsRegion userAgent errorCode eventType managementEvent readOnly eventCategory requestId

預設欄位索引是附加於您在政策中定義的任何自訂欄位索引。預設欄位索引不會計入您的欄位索引配額。

filterIndex 與篩選條件的比較

若要說明 filterIndex和 之間的差異filter，請考慮下列範例查詢。假設您已為四個日誌群組建立的欄位索引IPaddress，但不是為第五個日誌群組建立。下列使用 的查詢filterIndex會略過掃描沒有欄位索引的日誌群組。對於每個索引日誌群組，它會嘗試僅掃描具有索引欄位的日誌事件，而且也會在建立欄位索引之後僅傳回 的結果。

```
fields @timestamp, @message
| filterIndex IPaddress = "198.51.100.0"
| limit 20
```

相反地，如果您使用 filter而不是 filterIndex 來查詢相同的五個日誌群組，查詢不僅會嘗試掃描索引日誌群組中包含 值的日誌事件，還會掃描未編製索引的第五個日誌群組，而且會掃描該第五個日誌群組中的每個日誌事件。

```
fields @timestamp, @message
| filter IPaddress = "198.51.100.0"
| limit 20
```

SOURCE

當您使用 或 API 建立查詢時，在查詢SOURCE中包含 是指定要包含在查詢中的日誌群組和/ AWS CLI 或資料來源的實用方式。SOURCE 命令僅在 AWS CLI 和 API 中受支援，在 CloudWatch 主控台中不受支援。當您使用 CloudWatch 主控台啟動查詢時，您可以使用主控台界面來指定日誌群組。

查詢日誌群組

若要使用 SOURCE 指定要查詢的日誌群組，您可以使用下列關鍵字：

- namePrefix 針對名稱開頭為您所指定字串的日誌群組執行查詢。如果您省略此選項，則會查詢所有日誌群組。

  您可以在清單中包含最多五個字首。
- accountIdentifier 針對指定 AWS 帳戶中的日誌群組執行查詢。這只有在您在監控帳戶中執行查詢時才有效。如果您省略此選項，預設為查詢所有連結的來源帳戶和目前的監控帳戶。如需跨帳戶可觀測性的詳細資訊，請參閱 CloudWatch 跨帳戶可觀測性。

  您可以在清單中包含多達 20 個帳戶識別符。
- logGroupClass 針對指定日誌類別中的日誌群組執行查詢，可以是標準存取或不常存取。如果您省略此選項，則會使用標準日誌類別的預設值。如需日誌類別的詳細資訊，請參閱 日誌類別。

由於您可以指定大量日誌群組以這種方式查詢，因此建議您SOURCE僅在利用您已建立之欄位索引的查詢中使用 。如需在日誌群組中為欄位編製索引的詳細資訊，請參閱 建立欄位索引以改善查詢效能並減少掃描磁碟區

下列範例會選取帳戶中的所有日誌群組。如果這是監控帳戶，則會選取監控和所有來源帳戶的日誌群組。如果日誌群組總數超過 10，000，則您會看到錯誤，提示您使用不同的日誌群組選取方法來減少日誌群組數量。

```
SOURCE logGroups()
```

下列範例會選取111122223333來源帳戶中的日誌群組。如果您在 CloudWatch 跨帳戶可觀測性的監控帳戶中啟動查詢，預設會選取所有來源帳戶和監控帳戶中的日誌群組。

```
SOURCE logGroups(accountIdentifiers:['111122223333'])
```

下一個範例會根據名稱字首選取日誌群組。

```
SOURCE logGroups(namePrefix: ['namePrefix1', 'namePrefix2'])
```

下列範例會選取不常存取日誌類別中的所有日誌群組。如果您未包含識別class符，查詢只會套用至標準日誌類別中的日誌群組，這是預設值。

```
SOURCE logGroups(class: ['INFREQUENT_ACCESS'])
```

下一個範例會選取 111122223333 帳戶中以特定名稱字首開頭且位於標準日誌類別的日誌群組。命令中未提及 類別，因為 Standard 是預設的日誌類別值。

```
SOURCE logGroups(accountIdentifiers:['111122223333'], namePrefix: ['namePrefix1',
 'namePrefix2']
```

最後一個範例顯示如何搭配 SOURCE命令使用 `start-query` AWS CLI 命令。

```
aws logs start-query
--region us-east-1
--start-time 1729728200
--end-time 1729728215
--query-string "SOURCE logGroups(namePrefix: ['Query']) | fields @message | limit 5"
```

查詢資料來源

若要使用 SOURCE 指定要查詢的資料來源，您可以使用 dataSource關鍵字。您可以在清單中包含多達十個資料來源。

下列範例會選取amazon_vpc.flow資料來源。

```
SOURCE dataSource(['amazon_vpc.flow'])
```

下列範例會選取amazon_vpc.flow資料來源，並根據日誌群組名稱字首來限制日誌群組。

```
SOURCE dataSource(['amazon_vpc.flow']) logGroups(namePrefix: ['namePrefix1'])
```

pattern

使用 pattern 自動將您的日誌資料叢集化，以形成模式。

模式是指日誌欄位之間反覆出現的共同文字結構。您可以使用 pattern 來顯示新興趨勢、監控已知錯誤，以及識別經常發生或高成本的日誌行。CloudWatch Logs Insights 也提供主控台體驗，您可以用來尋找並進一步分析日誌事件中的模式。如需詳細資訊，請參閱模式分析。

由於 pattern命令會自動識別常見模式，因此您可以使用它做為搜尋和分析日誌的起點。您也可以將 pattern 與 filter、 parse 或 sort 命令搭配使用，在更多經過微調的查詢中識別模式。

模式命令輸入

pattern 命令需有以下任何一項輸入：@message 欄位、以 parse 命令建立的擷取欄位，或使用一或多個字串函數操控的字串。

如果 CloudWatch Logs 無法推斷動態權杖所代表的資料類型， 會顯示為 <Token-*number*>，而##指出此權杖相較於其他動態權杖，出現在模式中的位置。

動態字符的常見範例包括錯誤代碼、IP 地址、時間戳記和請求 IDs。

模式命令輸出

pattern 命令會產生以下輸出：

- @pattern：日誌事件欄位之間反覆出現的共同文字結構。模式內不同的欄位，例如請求 ID 或時間戳記，會以字符表示。如果 CloudWatch Logs 可以判斷動態字符代表的資料類型，則會將字符顯示為 <*string*-*number*>。##是字符所代表資料類型的描述。相較於其他動態字符，##會顯示此字符在模式中的顯示位置。

  CloudWatch Logs 會根據分析包含該名稱的日誌事件內容來指派名稱的字串部分。

  如果 CloudWatch Logs 無法推斷動態權杖所代表的資料類型， 會顯示為 <Token-*number*>，而##指出此權杖相較於其他動態權杖，在模式中顯示的位置。

  例如，[INFO] Request time: <Time-1> ms 是日誌訊息 [INFO] Request time: 327 ms 可能的輸出。
- @ratio：所選期間和指定日誌群組中，符合已識別模式的日誌事件比例。例如，如果選取的日誌群組和期間中有一半的日誌事件符合模式，@ratio 就會傳回 0.50

- @sampleCount：所選期間和指定日誌群組中，符合已識別模式的日誌事件數量。
- @severityLabel：日誌嚴重性或層級，指明日誌中的資訊類型，例如 Error、Warning、Info 或 Debug。

範例

以下命令會識別所選時間範圍內指定日誌群組中具有類似結構的日誌，並依模式和數量將其分組

```
pattern @message
```

pattern 命令可以與 filter 命令搭配使用

```
filter @message like /ERROR/
| pattern @message
```

pattern 命令可與 parse 和 sort 命令搭配使用

```
filter @message like /ERROR/
| parse @message 'Failed to do: *' as cause
| pattern cause
| sort @sampleCount asc
```

差異

將請求時段中找到的日誌事件與先前相同長度時段的日誌事件進行比較。如此一來，您可以尋找趨勢，並找出特定日誌事件是否為新的。

將修飾詞新增至diff命令，以指定您要與之比較的時段：

- diff 會將目前選取時間範圍中的日誌事件與前一個時間範圍的日誌事件進行比較。
- diff previousDay 會將目前選取時間範圍中的日誌事件與前一天同一時間的日誌事件進行比較。
- diff previousWeek 會將目前選取時間範圍中的日誌事件與上週同一時間的日誌事件進行比較。
- diff previousMonth 會將目前選取時間範圍中的日誌事件與上個月相同時間的日誌事件進行比較。

如需詳細資訊，請參閱比較（差異）與先前的時間範圍。

parse

使用 parse 從日誌欄位擷取資料，並建立一個您可在查詢中處理的擷取欄位。**parse** 支援使用萬用字元的 glob 模式和規則運算式。如需規則表達式語法的資訊，請參閱 支援的規則運算式 (regex) 語法。

您可以使用規則表達式剖析巢狀 JSON 欄位。

範例：剖析巢狀 JSON 欄位

程式碼片段會示範如何剖析在擷取期間已扁平化的 JSON 日誌事件。

```
{'fieldsA': 'logs', 'fieldsB': [{'fA': 'a1'}, {'fA': 'a2'}]}
```

程式碼片段會顯示一個具有規則運算式的查詢，其會擷取 fieldsA 和 fieldsB 的值，以建立擷取欄位 fld 和 array。

```
parse @message "'fieldsA': '*', 'fieldsB': ['*']" as fld, array
```

具名擷取群組

當您將 **parse** 與正規表達式搭配使用時，您可以使用具名擷取群組將模式擷取到欄位中。語法是 parse @message (?<*Name*>*pattern*)。

以下範例在 VPC 流量日誌上使用擷取群組，將 ENI 擷取到名為 NetworkInterface 的欄位中。

```
parse @message /(?<NetworkInterface>eni-.*?) / | display NetworkInterface, @message
```

> **Note**
>
> JSON 日誌事件會在擷取期間扁平化。目前不支援剖析具有 glob 表達式的巢狀 JSON 欄位。您只能剖析包含不超過 200 個日誌事件欄位的 JSON 日誌事件。剖析巢狀 JSON 欄位時，您必須格式化查詢中的規則表達式，以符合 JSON 日誌事件的格式。

剖析命令的範例

使用 glob 運算式，從日誌欄位 **@message** 中擷取欄位 **@user**、**@method** 和 **@latency**，並傳回 **@method** 和 **@user** 各種不重複組合的平均延遲。

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
```

```
    @user
```

使用規則運算式，從日誌欄位 **@message** 中擷取欄位 **@user2**、**@method2** 和 **@latency2**，並傳回 **@method2** 和 **@user2** 各種不重複組合的平均延遲。

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

擷取欄位 **loggingTime**、**loggingType** 和 **loggingMessage**，並篩選包含 **ERROR** 或 **INFO** 字串的日誌事件，然後針對包含 **ERROR** 字串的事件，僅顯示 **loggingMessage** 和 **loggingType** 欄位。

```
FIELDS @message
    | PARSE @message "* [*] *" as loggingTime, loggingType, loggingMessage
    | FILTER loggingType IN ["ERROR", "INFO"]
    | DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

sort

使用 `sort` 按指定欄位以遞增 (`asc`) 或遞減 (`desc`) 順序顯示日誌事件。您可以搭配使用 `limit` 命令來建立「前 N 個」或「底部 N 個」查詢。

排序演算法是自然排序的更新版本。如果您以遞增順序排序，則會使用下列邏輯。

- 所有非數值都先於所有數值。數值是僅包含數字的值，而不是數字和其他字元的組合。
- 對於非數值，演算法會將連續的數值字元和連續的字母字元分組為不同的區塊以供比較。它會依其 Unicode 值排序非數值部分，然後依其長度排序數值部分，再依其數值排序。

如需 Unicode 順序的詳細資訊，請參閱 Unicode 字元清單。

例如，以下是依遞增順序排序的結果。

```
!: >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> sorted by unicode order
#
*%04
0# >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Alphanumeric starting with numbers
5A
111A   >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>  Starts with more digits than 5A, so it sorted
 to be later than 5A
2345_
@ >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> 2345 is compared with @ in the unicode order,
```

```
@_
A >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Values starting with letters
A9876fghj
a12345hfh
0 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Number values
01
1
2
3
```

如果您以遞減順序排序，排序結果會相反。

例如，下列 Amazon VPC 流程日誌的查詢會找出跨主機的前 15 個封包傳輸。

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

統計資料

使用 stats 建立日誌資料的視覺化效果，例如長條圖、折線圖和堆疊區域圖。這可協助您更有效地識別日誌資料中的模式。CloudWatch Logs Insights 可以為使用 stats 函數和一或多個彙總函數的查詢產生視覺效果。

例如，Route 53 日誌群組中的下列查詢會傳回視覺化效果，依查詢類型顯示每小時 Route 53 記錄的分佈情況。

```
stats count(*) by queryType, bin(1h)
```

所有這些查詢都可以產生長條圖。如果您的查詢使用 bin() 函式將資料以一個欄位與一段時間進行群組，那麼您也可以看到折線圖和堆疊區域圖。

bin 函數支援以下時間單位和縮寫。對於包含多個字元的所有單位和縮寫，支援加上 s 來表示複數。所以 hr 和 hrs 皆可用來指定時數。

- millisecond ms msec
- second s sec
- minute m min
- hour h hr
- day d

- week w
- month mo mon
- quarter q qtr
- year y yr

主題

- 視覺化呈現時間序列資料
- 視覺化呈現依欄位分組的日誌資料
- 在單一查詢中使用多個統計資訊命令
- 與統計資料搭配使用的函數

視覺化呈現時間序列資料

時間序列視覺化適用於具有下列特性的查詢：

- 查詢包含一或多個彙總函數。如需詳細資訊，請參閱Aggregation Functions in the Stats Command。
- 查詢使用 bin() 函數依一個欄位來分組資料。

這些查詢可以產生折線圖、堆疊區域圖、長條圖和圓餅圖。

範例

如需完整的教學，請參閱the section called “教學課程：執行查詢來產生時間序列視覺化”。

以下是更多適用於時間序列視覺化的查詢範例。

以下查詢為 myfield1 欄位的平均值產生視覺效果，其中每 5 分鐘建立一個資料點。每個資料點是日誌中前五分鐘的 myfield1 值的平均值彙總。

```
stats avg(myfield1) by bin(5m)
```

以下查詢根據不同欄位建立三個值的視覺效果，其中每 5 分鐘建立一個資料點。產生此覺化是因為查詢包含彙總函數，且使用 bin() 做為分組欄位。

```
stats avg(myfield1), min(myfield2), max(myfield3) by bin(5m)
```

折線圖和堆疊區域圖限制

彙總記錄項目資訊但不使用 bin() 函數的查詢可產生長條圖。不過，該查詢無法產生折線圖或堆疊區域圖。如需這些查詢類型的詳細資訊，請參閱 the section called “視覺化呈現依欄位分組的日誌資料”。

視覺化呈現依欄位分組的日誌資料

您可以為使用 stats 函數和一或多個彙總函數的查詢產生長條圖。如需詳細資訊，請參閱Aggregation Functions in the Stats Command。

若要查看視覺化，請執行查詢。查詢 Visualization (視覺化) 標籤，選取 Line (線條) 旁邊的箭頭，然後選擇 Bar (長條)。長條圖中的視覺化限制為最多 100 個長條。

範例

如需完整的教學，請參閱the section called “教學課程：執行查詢以產生依日誌欄位分組的視覺效果”。以下段落包含更多可依據欄位進行視覺化的查詢範例。

下列 VPC 流程日誌查詢會尋找每個目的地位址、每個工作階段傳輸的平均位元組數。

```
stats avg(bytes) by dstAddr
```

您也可以產生一個圖表，其中包含每個結果值的多個長條。例如，下列 VPC 流程日誌查詢會尋找每個目的地位址、每個工作階段傳輸的平均和最大位元組數。

```
stats avg(bytes), max(bytes) by dstAddr
```

下列查詢會尋找每個查詢類型的 Amazon Route 53 查詢日誌數量。

```
stats count(*) by queryType
```

在單一查詢中使用多個統計資訊命令

您可以在單一查詢中使用多達兩個 stats 命令。這讓能您在第一個彙總的輸出上執行額外的彙總。

範例：使用兩個 **stats** 命令進行查詢

例如，以下查詢會先找出 5 分鐘區間的總流量，然後計算這些 5 分鐘區間的最高、最低和平均流量。

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length)/1024/1024 as logs_mb BY bin(5m)
| STATS max(logs_mb) AS peak_ingest_mb,
        min(logs_mb) AS min_ingest_mb,
```

```
        avg(logs_mb) AS avg_ingest_mb
```

範例：將多個統計資料命令與其他函數 (例如 **filter**、**fields**、**bin**) 相結合

您可以在單一命令中，將兩個 stats 命令與其他命令 (例如 filter 和 fields) 相結合。例如，以下查詢會尋找工作階段中不同 IP 地址的數目，並依用戶端平台尋找工作階段數目，篩選這些 IP 地址，最後再找出每個用戶端平台的工作階段請求的平均數。

```
STATS count_distinct(client_ip) AS session_ips,
      count(*) AS requests BY session_id, client_platform
| FILTER session_ips > 1
| STATS count(*) AS multiple_ip_sessions,
        sum(requests) / count(*) AS avg_session_requests BY client_platform
```

您可以在具有多個 stats 命令的查詢中使用 bin 和 dateceil 函數。例如，以下查詢會先將訊息合併成 5 分鐘的區塊，然後將這些 5 分鐘的區塊彙總為 10 分鐘的區塊，並計算每個 10 分鐘區塊內的最高、最低和平均流量。

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length) / 1024 / 1024 AS logs_mb BY BIN(5m) as @t
| STATS max(logs_mb) AS peak_ingest_mb,
        min(logs_mb) AS min_ingest_mb,
        avg(logs_mb) AS avg_ingest_mb BY dateceil(@t, 10m)
```

備註與限制

查詢最多可以有兩個 stats 命令。此配額無法變更。

如果您使用一個 sort 或 limit 命令，則其必須出現在第二個 stats 命令之後。如果在第二個 stats 命令之前，查詢無效。

當查詢有兩個 stats 命令時，在第一個 stats 彙總完成之前，不會開始顯示查詢的部分結果。

在單一查詢的第二個 stats 命令中，您只能參照第一個 stats 命令中定義的欄位。例如，以下查詢無效，因為 @message 欄位要在第一次 stats 彙總之後才可以使用。

```
FIELDS @message
| STATS SUM(Fault) by Operation
# You can only reference `SUM(Fault)` or Operation at this point
| STATS MAX(strlen(@message)) AS MaxMessageSize # Invalid reference to @message
```

您在第一個 stats 命令之後參照的任何欄位，都必須在該第一個 stats 命令中定義。

```
STATS sum(x) as sum_x by y, z
| STATS max(sum_x) as max_x by z
# You can only reference `max(sum_x)`, max_x or z at this point
```

Important

bin 函數始終以隱含的方式使用 @timestamp 欄位。這表示如果不使用第一個 stats 命令傳播 timestamp 欄位，就無法在第二個 stats 命令中使用 bin。例如，以下查詢無效。

```
FIELDS strlen(@message) AS message_length
 | STATS sum(message_length) AS ingested_bytes BY @logStream
 | STATS avg(ingested_bytes) BY bin(5m) # Invalid reference to @timestamp field
```

因此，應在第一個 stats 命令中定義 @timestamp 欄位，然後就可以在第二個 stats 命令中用來與 dateceil 搭配使用，如以下範例所示。

```
FIELDS strlen(@message) AS message_length
 | STATS sum(message_length) AS ingested_bytes, max(@timestamp) as @t BY
 @logStream
 | STATS avg(ingested_bytes) BY dateceil(@t, 5m)
```

與統計資料搭配使用的函數

CloudWatch Logs Insights 支援統計資料彙總函數和統計資料非彙總函數。

在 stats 命令中使用統計資料彙總函數，並用作其他函數的引數。

函式	結果類型	Description
avg(fieldName: NumericLogField)	number	所指定欄位中的值的平均數。
count() count(fieldName: LogField)	number	計算日誌事件數。count() (或 count(*)) 計算查詢傳回的所有事件數，count(fieldName) 計算包含指定欄位名稱的所有記錄數。

函式	結果類型	Description
`count_distinct(fieldName: LogField)`	number	傳回欄位的唯一值數目。如果欄位有極高的基數 (包含許多唯一值)，則 `count_distinct` 傳回的值只是近似值。
`max(fieldName: LogField)`	LogFieldValue	在所查詢的日誌中此日誌欄位的值上限。
`min(fieldName: LogField)`	LogFieldValue	在所查詢的日誌中此日誌欄位的值下限。
`pct(fieldName: LogFieldValue, percent: number)`	LogFieldValue	百分位數會指出資料集中相關準備好的值。例如，`pct(@duration, 95)` 傳回 `@duration` 值，其中 `@duration` 的值有 95% 低於這個值，有 5% 高於這個值。
`stddev(fieldName: NumericLogField)`	number	所指定欄位中的值的標準差。
`sum(fieldName: NumericLogField)`	number	所指定欄位中的值的總和。

Stats 非彙總函數

非彙總函數可用於 `stats` 命令，也可以作為其他函數的引數使用。

函式	結果類型	Description
`earliest(fieldName: LogField)`	LogField	從在查詢日誌中具有最早時間戳記的日誌事件中傳回 `fieldName` 的值。
`latest(fieldName: LogField)`	LogField	從在查詢日誌中具有最晚時間戳記的日誌事件中傳回 `fieldName` 的值。
`sortsFirst(fieldName: LogField)`	LogField	傳回在查詢日誌中最先排序的 `fieldName` 值。

函式	結果類型	Description
`sortsLast(fieldName: LogField)`	LogField	傳回在查詢日誌中最後排序的 `fieldName` 值。

limit

使用 `limit` 指定您希望查詢傳回的日誌事件數目。如果您省略 `limit`，查詢會在結果中傳回多達 10，000 個日誌事件。

例如，下列範例僅傳回 25 個最新的日誌事件。

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

dedup

使用 dedup 根據指定欄位中的特定值移除重複的結果。可以將 dedup 與一個或多個欄位搭配使用。如果對 dedup 指定一個欄位，則只會針對該欄位的每個唯一值傳回一個日誌事件。如果指定多個欄位，則會針對這些欄位的每個唯一值組合傳回一個日誌事件。

系統會根據排序順序捨棄重複項目，只會保留排序順序中的第一個結果。建議您先對結果進行排序，然後再透過 dedup 命令進行排序。如果在透過 dedup 執行之前未對結果進行排序，則會採用使用 `@timestamp` 的預設遞減排序順序。

Null 值不會被視為評估的重複項目。系統會保留任何指定欄位之具有 Null 值的日誌事件。要消除具有 null 值的字段，請採用使用 `isPresent(field)` 函數的 **`filter`**。

可以在 dedup 命令之後的查詢中使用的唯一查詢命令為 `limit`。

當您在查詢dedup中使用 時，主控台會顯示訊息，例如顯示 Y 記錄的 X，其中 X 是重複資料刪除的結果數目，Y 是重複資料刪除之前符合的記錄總數。這表示重複的記錄已移除，並不表示資料遺失。

範例：僅查看名為 **`server`** 之欄位的每個唯一值的最近日誌事件

下列範例顯示 `server` 的每個唯一值的最近事件的 `timestamp`、`server`、`severity` 和 `message` 欄位。

```
fields @timestamp, server, severity, message
| sort @timestamp desc
```

```
| dedup server
```

如需 CloudWatch Logs Insights 查詢的更多範例，請參閱 一般查詢。

unmask

使用 unmask 可以顯示因為資料保護政策而遮罩某些內容的某個日誌事件的全部內容。若要使用此命令，您必須擁有 logs:Unmask 許可。

如需有關日誌群組中資料保護的詳細資訊，請參閱 使用遮罩功能協助保護敏感日誌資料。

解巢狀

使用 unnest將做為輸入的清單扁平化，為清單中的每個元素產生具有單一記錄的多個記錄。根據欄位包含的項目數量，此命令會捨棄目前的記錄並產生新的記錄。每個記錄都包含 unnested_field，代表一個項目。所有其他欄位都來自原始記錄。

的輸入unnest是 LIST，其來自 jsonParse函數。如需詳細資訊，請參閱結構類型。任何其他類型，例如 MAPString和 numbers，都會被視為清單中有一個項目unnest。

命令結構

下列範例說明此命令的格式。

```
unnest field into unnested_field
```

查詢範例

下列範例剖析 JSON 物件字串並展開欄位事件清單。

```
fields jsonParse(@message) as json_message
| unnest json_message.events into event
| display event.name
```

此範例查詢的日誌事件可以是 JSON 字串，如下所示：

```
{
   "events": [
        {
            "name": "exception"
```

```
        },
        {
            "name": "user action"
        }
    ]
}
```

在此情況下，範例查詢會在查詢結果中產生兩個記錄，一個為 event.name，exception另一個為 event.name作為使用者動作

查詢範例

下列範例會扁平化清單，然後篩選出項目。

```
fields jsonParse(@message) as js
| unnest js.accounts into account
| filter account.type = "internal"
```

查詢範例

下列範例會扁平化要彙總的清單。

```
fields jsonParse(trimmedData) as accounts
| unnest accounts into account
| stats sum(account.droppedSpans) as n by account.accountId
| sort n desc
| limit 10
```

查詢

使用 lookup 以查詢資料表中的參考資料豐富查詢結果。查詢資料表包含您上傳至 Amazon CloudWatch Logs 的 CSV 資料。查詢執行時，lookup命令會將日誌事件中的欄位與查詢資料表中的欄位相符，並將指定的輸出欄位附加至結果。

針對資料擴充案例使用查詢表，例如將使用者 IDs映射至使用者詳細資訊、將產品代碼映射至產品資訊，或將錯誤代碼映射至錯誤描述。

建立和管理查詢資料表

您必須先建立查詢資料表，才能在查詢中使用 lookup命令。您可以從 CloudWatch 主控台或使用 Amazon CloudWatch Logs API 建立和管理查詢資料表。

建立查詢資料表（主控台）

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇設定，然後選擇日誌索引標籤。
3. 捲動至查詢資料表，然後選擇管理。
4. 選擇建立查詢資料表。
5. 輸入查詢資料表的名稱。名稱只能包含英數字元、連字號和底線。
6. (選用) 輸入描述。
7. 上傳 CSV 檔案。檔案必須包含具有資料欄名稱的標頭列、使用 UTF-8 編碼，且不得超過 10 MB。
8. （選用）指定要加密資料表資料的 AWS KMS 金鑰。
9. 選擇建立。

建立查詢資料表後，您可以在 CloudWatch Logs Insights 查詢編輯器中檢視它。選擇查詢資料表索引標籤來瀏覽可用的資料表及其欄位。

若要更新查詢資料表，請選取資料表，然後選擇動作、更新。上傳新的 CSV 檔案以取代所有現有的內容。若要刪除查詢資料表，請選擇動作、刪除。

> **Note**
>
> 每個 每個帳戶最多可以建立 100 個查詢資料表 AWS 區域。CSV 檔案最多可達 10 MB。您也可以使用 Amazon CloudWatch Logs API 管理查詢資料表。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 CreateLookupTable。

> **Note**
>
> 如果查詢表使用 KMS 金鑰加密，呼叫者必須擁有金鑰（用於加密查詢表的 KMS 金鑰）的`kms:Decrypt`許可，才能搭配參考該查詢表的查詢使用 `StartQuery` API。如需詳細資訊，請參閱使用 加密 CloudWatch Logs 中的查詢資料表 AWS Key Management Service。

查詢查詢語法

命令結構

以下顯示此命令的格式。

```
lookup table lookup-field as log-field [,...] output-mode output-field[,...]
```

命令使用下列引數：

- *table* – 要使用的查詢資料表名稱。
- *lookup-field* – 查詢資料表中要比對的欄位。
- *log-field* – 日誌事件中要比對的欄位。相符項目完全且區分大小寫。
- *output-mode* – 指定 OUTPUT將輸出欄位新增至結果。如果日誌事件中已存在同名的欄位，則會予以覆寫。
- *output-field* – 要新增至結果的查詢資料表中的一個或多個欄位。

範例：使用使用者詳細資訊來豐富日誌事件

假設您有一個包含 id 欄位的事件日誌群組，以及名為 的查詢資料表user_data，其中包含欄 id、name、 email和 department。下列查詢會使用查詢表中的使用者名稱、電子郵件和部門來豐富每個日誌事件。

```
fields action, status, name, email, department
| lookup user_data id OUTPUT name, email, department
```

範例：搭配彙總使用查詢

您可以搭配彙總函數使用查詢輸出欄位。下列查詢會使用使用者詳細資訊豐富日誌事件，然後計算依電子郵件地址分組的事件。

```
fields user_id, action, username, email, department
| lookup user_data user_id OUTPUT username, email, department
| stats count(*) by email
```

範例：搭配篩選條件使用查詢

您可以根據查詢傳回的欄位來篩選結果。下列查詢會擴充日誌事件，然後篩選以僅顯示特定部門的事件。

```
fields user_id, action
| lookup user_data user_id OUTPUT username, email, department
```

```
| filter department = "Engineering"
```

布林值、比較、數值、日期時間和其他函數

CloudWatch Logs Insights 支援查詢中的許多其他操作和函數，如以下各節所述。

主題

算術運算子

算術運算子可接受以數值資料類型作為引數，而且會傳回數值結果。算術運算子可用於 `filter` 和 `fields` 命令，也可以作為其他函數的引數使用。

作業	Description
`a + b`	加法
`a - b`	減法
`a * b`	乘法
`a / b`	除法
`a ^ b`	指數 (`2 ^ 3` 傳回 8)
`a % b`	餘數或模數 (`10 % 3` 傳回 1)

布林值運算子

使用布林值運算子 **and**、**or** 和 **not**。

> **Note**
>
> 布林值運算子僅限用於會傳回 TRUE 或 FALSE 的函數。

比較運算子

比較運算子可接受以所有資料類型作為引數，而且會傳回布林值結果。比較運算子可用於 `filter` 命令，也可以作為其他函數的引數使用。

運算子	Description
=	等於
!=	不等於
<	Less than
>	Greater than
<=	小於或等於
>=	大於或等於

數值運算子

數值運算接受數值資料類型作為引數，並傳回數值結果。數值運算可用於 `filter` 和 `fields` 命令，也可以作為其他函數的引數使用。

作業	結果類型	Description
`abs(a: number)`	number	絕對值
`ceil(a: number)`	number	無條件進位到上限 (大於 a 值的最小整數)

作業	結果類型	Description
`floor(a: number)`	number	無條件捨去到下限 (小於 a 值的最大整數)
`greatest(a: number, ...numbers: number[])`	number	傳回最大值
`least(a: number, ...numbers: number[])`	number	傳回最小值
`log(a: number)`	number	自然對數
`sqrt(a: number)`	number	平方根

結構類型

地圖或清單是 CloudWatch Logs Insights 中的結構類型，可讓您存取和使用查詢的屬性。

範例：取得地圖或清單

使用 `jsonParse`將 json 字串的欄位剖析為映射或清單。

```
fields jsonParse(@message) as json_message
```

範例：存取屬性

使用點存取運算子 (map.attribute) 來存取映射中的項目。如果地圖中的屬性包含特殊字元，請使用反引號括住屬性名稱 (map.attributes.`special.char`)。

```
fields jsonParse(@message) as json_message
| stats count() by json_message.status_code
```

使用括號存取運算子 (list【index】) 擷取清單中特定位置的項目。

```
fields jsonParse(@message) as json_message
| filter json_message.users[1].action = "PutData"
```

當金鑰名稱中存在特殊字元時，以反引號 (``) 包裝特殊字元。

```
fields jsonParse(@message) as json_message
| filter json_message.`user.id` = "123"
```

範例：空的結果

地圖和清單會被視為字串、數字和日期時間函數的 null。

```
fields jsonParse(@message) as json_message
| display toupper(json_message)
```

比較映射和清單到任何其他欄位會導致 `false`。

> **Note**
>
> `stats` 不支援在 `dedup`、`sort`、 和 `pattern`中使用地圖和清單。

日期時間函數

日期時間函數

日期時間函數可用於 `fields` 和 `filter` 命令，也可以作為其他函數的引數使用。如果查詢中使用了彙總函數，您可以使用這些函數來建立時段。使用由數字和下列其中一項組成的時段：

- `ms` 毫秒
- `s` 持續 秒
- `m` 持續 分鐘
- `h` 小時

例如，`10m` 是 10 分鐘，`1h` 是 1 小時。

> **Note**
>
> 為您的日期時間函數使用最適當的時間單位。CloudWatch Logs 會根據您選擇的時間單位來限制您的請求。例如，它以 60 為上限，作為使用 的任何請求的最大值`s`。因此，如果您指定 `bin(300s)`，CloudWatch Logs 實際上會將其實作為 60 秒，因為 60 是一分鐘的秒

數，因此 CloudWatch Logs 不會搭配 使用高於 60 的數字s。若要建立 5 分鐘的儲存貯體，請bin(5m)改用 。
的上限為 ms 1000、 s和 的上限為 m 60，而 的上限為 h 24。

下表列出您可以在查詢命令中使用的不同日期時間函數。該表列出了每個函式的結果類型，並包含對每個函式的說明。

> **Tip**
>
> 建立查詢命令時，您可以使用時間間隔選擇器，來選取您要查詢的時間段。例如：您可以設定 5 分鐘到 30 分鐘的間隔；1 小時、3 小時和 12 小時的間隔；或是自訂的時間範圍。您也可以在特定日期之間設定時間段。

函式	結果類型	Description

`bin(period: Period)`	時間戳記	將 `@timestamp` 的值四捨五入到指定時間段，然後截斷。例如，`bin(5m)` 將 `@timestamp` 的值四捨五入至最接近的 5 分鐘。
		
		您可以使用此操作在查詢中將多筆日誌條目分組在一起。以下範例傳回每小時的例外情況計數。
		
		`filter @message like /Exception/`
		`  | stats count(*) as exceptionCount`
		`by bin(1h)`
		`  | sort exceptionCount desc`
		
		bin 函數支援以下時間單位和縮寫。對於包含多個字元的所有單位和縮寫，支援加上 s 來表示複數。所以 hr 和 hrs 皆可用來指定時數。
		
		• millisecond ms msec
		• second s sec
		• minute m min
		• hour h hr

函式	結果類型	Description
		• day d • week w • month mo mon • quarter q qtr • year y yr
datefloor(timestamp: Timestamp, period: Period)	時間戳記	將時間戳記截斷為指定的期間。例如，datefloor(@timestamp, 1h) 將 @timestamp 的所有值截斷為半點小時。
dateceil(timestamp : Timestamp, period: Period)	時間戳記	將時間戳記無條件進位到指定期間，然後截斷。例如，dateceil(@timestamp, 1h) 將 @timestamp 的所有值截斷為整點小時。
fromMillis(fieldNa me: number)	時間戳記	解譯輸入欄位為自 Unix epoch 以來的毫秒數，並將其轉換為時間戳記。
toMillis(fieldName: Timestamp)	number	將指定欄位中找到的時間戳記轉換為數字，代表自 Unix epoch 以來的毫秒數。例如：toMillis(@timestamp) 會將時間戳記 2022-01-14T13:18:031.000-08:00 轉換為 1642195111000 。

函式	結果類型	Description

now( )	number	以 epoch 秒為單位，傳回查詢處理開始的時間。此函數不採用引數。
		
		您可以使用此值，根據目前時間篩選查詢結果。
		
		例如，下列查詢會傳回過去兩個小時內所有 4xx 個錯誤：
		
		`parse @message "Status Code: *;" as statusCode\n`
		`| filter statusCode >= 400 and statusCode <= 499 \n`
		`| filter toMillis(@timestamp) >= (now() * 1000 - 7200000)`
		
		下列範例會傳回過去五小時內包含單字 `error`或 的所有日誌項目 `failure`
		
		`fields @timestamp, @message`
		`| filter @message like /(?i)(error|failure)/`
		`| filter toMillis(@timestamp) >= (now() * 1000 - 18000000)`

> **Note**
>
> 目前 CloudWatch Logs Insights 不支援使用人類可讀時間戳記篩選日誌。

一般函數

一般函數

一般函數可用於 `fields` 和 `filter` 命令，也可以作為其他函數的引數使用。

函式	結果類型	Description
`ispresent(fieldName: LogField)`	Boolean	如果欄位存在，傳回 `true`
`coalesce(fieldName: LogField, ...fieldNames: LogField[])`	LogField	傳回清單中的第一個非空值

JSON 函數

JSON 函數

在 `fields`和 `filter`命令中使用 JSON 函數，並做為其他函數的引數。

函式	結果類型	Description
`jsonParse(fieldName: string)`	映射 | 清單 | 空白	當輸入是 JSON 物件或 JSON 陣列的字串表示法時，傳回映射或清單。如果輸入不是其中一個表示法，則傳回空值。
`jsonStringify(fieldName: Map | List)`	String	從映射或清單資料傳回 JSON 字串。

IP 地址字串函數

IP 地址字串函數

IP 地址字串函數可用於 `filter` 和 `fields` 命令，也可以作為其他函數的引數使用。

函式	結果類型	Description
`isValidIp(fieldName: string)`	boolean	如果欄位是有效的 IPv4 或 IPv6 地址，則會傳回 `true`。
`isValidIpV4(fieldName: string)`	boolean	如果欄位是有效的 IPv4 地址，則傳回 `true`。

函式	結果類型	Description
`isValidIpV6(fieldName: string)`	boolean	如果欄位是有效的 IPv6 地址，則傳回 `true`。
`isIpInSubnet(fieldName: string, subnet: string)`	boolean	如果欄位是指定 v4 或 v6 子網路內的有效 IPv4 或 IPv6 地址，則傳回 `true`。指定子網路時，請使用 CIDR 標記法，例如 `192.0.2.0/24` 或 `2001:db8::/32` ，其中 `192.0.2.0` 或 `2001:db8::` 是 CIDR 區塊的起始位置。
`isIpv4InSubnet(fieldName: string, subnet: string)`	boolean	如果欄位是指定 v4 子網路內的有效 IPv4 地址，則傳回 `true`。指定子網路時，請使用 CIDR 標記法，例如 `192.0.2.0/24` ，其中 `192.0.2.0` 是 CIDR 區塊的起始位置。
`isIpv6InSubnet(fieldName: string, subnet: string)`	boolean	如果欄位是指定 v6 子網路內的有效 IPv6 地址，則傳回 `true`。指定子網路時，請使用 CIDR 標記法，例如 `2001:db8::/32` ，其中 `2001:db8::` 是 CIDR 區塊的起始位置。

字串函數

字串函數

字串函數可用於 `fields` 和 `filter` 命令，也可以作為其他函數的引數使用。

函式	結果類型	Description
`isempty(fieldName: string)`	Number	如果欄位遺失或是空白字串，傳回 1。
`isblank(fieldName: string)`	Number	如果欄位遺失、是空白字串或只包含空格，傳回 1。
`concat(str: string, ...strings: string[])`	string	串連字串。

函式	結果類型	Description

`ltrim(str: string)`
 	
`ltrim(str: string, trimChars: string)`
	string	如果函數沒有第二個引數，則會移除字串左側的空格。如果函數有第二個字串引數，則不會移除空格。而是從 `str` 左側移除 `trimChars` 中的字元。例如，`ltrim("xyZxyfooxyZ","xyZ")` 傳回 `"fooxyZ"`。

`rtrim(str: string)`
 	
`rtrim(str: string, trimChars: string)`
	string	如果函數沒有第二個引數，則會移除字串右側的空格。如果函數有第二個字串引數，則不會移除空格。而是從 `str` 右側移除 `trimChars` 的字元。例如，`rtrim("xyZfooxyxyZ","xyZ")` 傳回 `"xyZfoo"`。

`trim(str: string)`
 	
`trim(str: string, trimChars: string)`
	string	如果函數沒有第二個引數，則會移除字串兩側的空格。如果函數有第二個字串引數，則不會移除空格。而是從 `str` 兩側移除 `trimChars` 的字元。例如，`trim("xyZxyfooxyxyZ","xyZ")` 傳回 `"foo"`。

`strlen(str: string)`	number	以 Unicode 字碼指標傳回字串的長度。
`toupper(str: string)`	string	將字串轉換成大寫。
`tolower(str: string)`	string	將字串轉換成小寫。

函式	結果類型	Description

`substr(str: string, startIndex: number)`
 	
`substr(str: string, startIndex: number, length: number)`
	string	傳回從數字引數指定的索引到字串結尾的子字串。如果函數有第二個數字引數，則是包含要擷取的子字串長度。例如，`substr("xyZfooxyZ",3, 3)` 傳回 `"foo"`。

`replace(fieldName: string, searchValue: string, replaceValue: string)`	string	以 `replaceValue` 取代 `fieldName: string` 中出現的所有 `searchValue` 。 例如：函式 `replace(logGroup,"smoke_test","Smoke")` 搜尋欄位 `logGroup` 中包含字串值 `smoke_test` 的日誌事件，並將值替換為字串 `Smoke`。
`strcontains(str: string, searchValue: string)`	number	如果 `str` 包含 `searchValue` ，則傳回 1，否則傳回 0。

包含特殊字元的欄位

如果欄位包含@符號或句點 (.) 以外的非英數字元，您必須以反引號字元 () 括住欄位`。例如：日誌欄位 `foo-bar` 含有非英數字元，亦即連字號 (-)，因此必須置於反引號 (`foo-bar`) 之間。

在查詢中使用別名和註解

建立含有別名的查詢。將日誌欄位重新命名，或在擷取值並填入欄位時，都可使用別名。使用關鍵字 `as` 為日誌欄位或結果賦予別名。您可以在查詢中使用多個別名。您可以在下列任一命令中使用別名：

- `fields`
- `parse`
- `sort`
- `stats`

以下範例會示範如何建立含有別名的查詢。

範例

查詢的 fields 命令中含有別名。

```
fields @timestamp, @message, accountId as ID
| sort @timestamp desc
| limit 20
```

查詢會傳回欄位 @timestamp、@message 和 accountId 的值。結果以遞減方式排序，且限制為 20。accountId 的值會顯示於別名 ID 底下。

範例

查詢的 sort 和 stats 命令中含有別名。

```
stats count(*) by duration as time
| sort time desc
```

查詢會計算欄位 duration 出現於日誌群組中的次數，並以遞減方式將結果排序。duration 的值會顯示於別名 time 底下。

使用註解

CloudWatch Logs Insights 支援在查詢中使用註解。使用雜湊字元 (#) 作為註解的開頭。您可以使用註解，忽略查詢或文件查詢中的行。

範例：查詢

以下查詢運行時，系統會忽略第二行。

```
fields @timestamp, @message, accountId
# | filter accountId not like "7983124201998"
| sort @timestamp desc
| limit 20
```

Logs Insights QL 入門：查詢教學課程

下列各節包含範例查詢教學課程，協助您開始使用 Logs Insights QL。

主題

教學課程：執行和修改範例查詢

下列教學課程協助您開始使用 CloudWatch Logs Insights。您可以在 Logs Insights QL 中執行範例查詢，然後查看如何修改並重新執行。

若要執行查詢，您必須已經有日誌存放在 CloudWatch Logs。如果您已經在使用 CloudWatch Logs，且已設定日誌群組和日誌串流，那就可以開始。如果您使用 AWS CloudTrail、Amazon Route 53 或 Amazon VPC 等服務，而且已設定將來自這些服務的日誌傳到 CloudWatch Logs，則您也可能已經有日誌。如需有關將日誌傳送至 CloudWatch Logs 的詳細資訊，請參閱開始使用 CloudWatch Logs。

CloudWatch Logs Insights 中的查詢會從日誌事件傳回一組欄位，或在日誌事件上執行的數學加總或其他運算的結果。本教學課程示範的查詢會傳回日誌事件清單。

執行範例查詢

執行 CloudWatch Logs Insights 範例查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。

   在 Logs Insights 頁面上，查詢編輯器包含 Logs Insights QL 中的預設查詢，該查詢會傳回 20 個最新的日誌事件。
3. 在 Select log group(s) (選取日誌群組) 下拉式選單中，選擇一個或多個要查詢的日誌群組。

   如果這是 CloudWatch 跨帳戶觀察功能中的監控帳戶，您可以在來源帳戶和監控帳戶中選取日誌群組。單一查詢可以一次查詢來自不同帳戶的日誌。

   您可以依日誌群組名稱、帳戶 ID 或帳戶標籤來篩選日誌群組。

   當您在標準日誌類別中選取日誌群組時，CloudWatch Logs Insights 會自動偵測群組中的資料欄位。若要查看探索的欄位，請選取頁面右上方附近的 Fields (欄位) 選單。

> **Note**
>
> 僅標準日誌類別中的日誌群組支援探索的欄位。如需日誌類別的詳細資訊，請參閱 日誌類別。

4. (選用) 使用時間間隔選擇器，選取您要查詢的時間段。

   您可以選擇 5 分鐘到 30 分鐘的間隔；1 小時、3 小時和 12 小時的間隔；或是自訂的時間範圍。

5. 選擇 Run (執行) 以檢視結果。

   在本教學中，結果包括 20 筆最近新增的日誌事件。

   CloudWatch Logs 會顯示日誌群組中一段時間內的日誌事件長條圖。長條圖不僅會顯示表格中的事件，還會顯示日誌群組中符合您的查詢和時間範圍的事件分佈。

6. 若要查看傳回日誌事件的所有欄位，請選擇編號事件左側的三角形下拉圖示。

修改範例查詢

在此教學課程中，您將修改範例查詢來顯示 50 個最新的日誌事件。

如果您尚未執行上一個教學課程，請現在這樣做。此教學課程會從上一個教學課程的結尾處開始。

> **Note**
>
> CloudWatch Logs Insights 隨附的一些範例查詢使用 `head` 或 `tail` 命令，而不是 `limit`。這些命令已被取代，並換成 `limit`。在您編寫的所有查詢中使用 `limit`，而不是 `head` 或 `tail`。

修改 CloudWatch Logs Insights 範例查詢

1. 在查詢編輯器中，將 20 變更為 50，然後選擇 Run (執行)。

   新查詢的結果隨即出現。假設日誌群組中有足夠的資料在預設時間範圍內，則現在會列出 50 個日誌事件。

2. (選用) 您可以儲存已建立的查詢。若要儲存此查詢，請選擇 Save (儲存)。如需詳細資訊，請參閱儲存並重新執行 CloudWatch Logs Insights 查詢。

將篩選條件命令新增到範例查詢

本教學課程說明如何在查詢編輯器對查詢進行更強大的變更。在此教學課程中，您將根據已擷取的日誌事件中的欄位，以篩選前一個查詢的結果。

如果您尚未執行先前的教學課程，請現在這樣做。此教學課程會從上一個教學課程的結尾處開始。

將篩選條件命令新增到前一個查詢

1. 決定要篩選的欄位。若要查看 CloudWatch Logs 過去 15 分鐘內在選定日誌群組包含的日誌事件中最常偵測到的欄位，以及每個欄位出現在這些日誌事件中的百分比，請在頁面右側選取 Fields (欄位)。

   若要查看特定日誌事件中包含的欄位，請選擇該列左側的圖示。

   awsRegion 欄位可能出現在您的日誌事件中，這取決於日誌中有哪些事件。在本教學剩下的部分，我們將使用 awsRegion 作為篩選條件欄位，但如果沒有該欄位，您可以使用不同的欄位。
2. 在查詢編輯器方塊中，將游標移到 50 後面，然後按 Enter。
3. 在新的一行上，首先輸入 | (垂直線字元) 和空格。CloudWatch Logs Insights 查詢中的命令必須以垂直線字元隔開。
4. 輸入 **`filter awsRegion="us-east-1"`**。
5. 選擇執行。

   查詢會再次執行，現在會顯示符合新篩選條件的 50 個最新結果。

   如果您篩選不同的欄位，且得到錯誤結果，則可能需要逸出欄位名稱。如果欄位名稱包含非英數字元，您必須在欄位名稱前後加上反引號字元 (`) (例如，**`` `error-code`="102" ``**)。

   您必須將反引號字元用於包含非英數字元的欄位名稱 ，而不是用於值。值一律包含在引號 (") 中。

Logs Insights QL 包含強大的查詢功能，包括對規則表達式、數學和統計操作的數個命令和支援。如需詳細資訊，請參閱CloudWatch Logs Insights 語言查詢語法。

教學課程：使用彙總函數執行查詢

您可以在 `stats` 命令中使用彙總函式，也可以作為其他函式的引數。在本教學中，您會了解如何執行查詢命令，以計算包含指定欄位的日誌事件數量。查詢命令會回傳按照指定欄位的一個或多個值分組的總數。如需有關彙總函式的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的支援的運算和函式。

使用彙總函式執行查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 確認已選取 Logs Insights QL 標籤。
4. 在 Select log group(s) (選取日誌群組) 下拉式選單中，選擇一個或多個要查詢的日誌群組。

   如果這是 CloudWatch 跨帳戶觀察功能中的監控帳戶，您可以在來源帳戶和監控帳戶中選取日誌群組。單一查詢可以一次查詢來自不同帳戶的日誌。

   您可以依日誌群組名稱、帳戶 ID 或帳戶標籤來篩選日誌群組。

   當您選取日誌群組時，如果日誌群組是標準類別日誌群組，CloudWatch Logs Insights 會自動偵測日誌群組中的資料欄位。若要查看探索的欄位，請選取頁面右上方附近的 Fields (欄位) 選單。
5. 刪除查詢編輯器中的預設查詢，然後輸入下列命令：

   ```
   stats count(*) by fieldName
   ```
6. 將 *fieldName* 替換為 Fields (欄位) 選單中探索到的選單欄位。

   Fields (欄位) 選單位於頁面右上角，會顯示 CloudWatch Logs Insights 在日誌群組中探索到的所有欄位。
7. 選擇 Run (執行) 以檢視查詢結果。

   查詢結果會顯示日誌群組中與查詢命令相符的記錄筆數，以及按照指定欄位的一個或多個值分組的總數。

教學課程：執行查詢以產生依日誌欄位分組的視覺效果

當您執行的查詢使用 `stats` 函數，依日誌項目中一或多個欄位的值來分組傳回的結果時，您可以透過長條圖、圓餅圖、折線圖或堆疊區域圖來檢視結果。這可協助您更有效率地將日誌中的趨勢視覺化。

執行查詢來產生視覺效果

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 在 Select log group(s) (選取日誌群組) 下拉式選單中，選擇一個或多個要查詢的日誌群組。

如果這是 CloudWatch 跨帳戶觀察功能中的監控帳戶，您可以在來源帳戶和監控帳戶中選取日誌群組。單一查詢可以一次查詢來自不同帳戶的日誌。

您可以依日誌群組名稱、帳戶 ID 或帳戶標籤來篩選日誌群組。

4. 在查詢編輯器中，刪除目前的內容，然後輸入以下 `stats` 函式，並選擇 Run query (執行查詢)。

```
stats count(*) by @logStream
    | limit 100
```

結果會顯示每個記錄串流的日誌群組中的日誌事件數量。結果限制為 100 個資料列。

5. 選擇 Visualization (視覺化) 標籤。
6. 選取 Line (行) 旁邊的箭頭，然後選擇 Bar (列)。

此時將會顯示長條圖，顯示日誌群組中每個日誌串流的長條圖。

教學課程：執行查詢來產生時間序列視覺化

當您執行的查詢使用 `bin()` 函數，依時段來分組傳回的結果時，您可以透過折線圖、堆疊區域圖、圓餅圖或長條圖來檢視結果。這可協助您更有效率地視覺化日誌事件在一段時間內的趨勢。

執行查詢來產生視覺效果

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 確認已選取 Logs Insights QL 標籤。
4. 在 Select log group(s) (選取日誌群組) 下拉式選單中，選擇一個或多個要查詢的日誌群組。

如果這是 CloudWatch 跨帳戶觀察功能中的監控帳戶，您可以在來源帳戶和監控帳戶中選取日誌群組。單一查詢可以一次查詢來自不同帳戶的日誌。

您可以依日誌群組名稱、帳戶 ID 或帳戶標籤來篩選日誌群組。

5. 在查詢編輯器中，刪除目前的內容，然後輸入以下 `stats` 函式，並選擇 Run query (執行查詢)。

```
stats count(*) by bin(30s)
```

結果會顯示 CloudWatch Logs 在每個 30 秒週期內在日誌群組中收到的日誌事件數。

6. 選擇 Visualization (視覺化) 標籤。

結果會顯示為折線圖。若要切換至長條圖、圓餅圖或堆疊區域圖，請選擇圖表左上角 Line (線條) 旁邊的箭頭。

範例查詢

本節列出您可以在CloudWatch 主控台執行的通用且有效的查詢命令。如需如何執行查詢命令的資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的教學：執行和修改範例查詢。

如需查詢語法的詳細資訊，請參閱 CloudWatch Logs Insights 語言查詢語法。

主題

- 一般查詢
- Lambda 日誌的查詢
- Amazon VPC 流程日誌的查詢
- Route 53 日誌的查詢
- CloudTrail 日誌的查詢
- 的查詢 Amazon API Gateway
- NAT 閘道的查詢
- Apache 伺服器日誌的查詢
- Amazon EventBridge 的查詢
- 剖析命令的範例

一般查詢

尋找最近新增的 25 個日誌事件。

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

取得每小時的例外狀況數清單。

```
filter @message like /Exception/
    | stats count(*) as exceptionCount by bin(1h)
    | sort exceptionCount desc
```

取得非例外狀況的日誌事件清單。

```
fields @message | filter @message not like /Exception/
```

取得 **server** 欄位的每個唯一值的最近日誌事件。

```
fields @timestamp, server, severity, message
| sort @timestamp asc
| dedup server
```

取得每個 **severity** 類型的 **server** 欄位的每個唯一值的最近日誌事件。

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server, severity
```

Lambda 日誌的查詢

查明過度佈建的記憶體數量。

```
filter @type = "REPORT"
    | stats max(@memorySize / 1000 / 1000) as provisonedMemoryMB,
        min(@maxMemoryUsed / 1000 / 1000) as smallestMemoryRequestMB,
        avg(@maxMemoryUsed / 1000 / 1000) as avgMemoryUsedMB,
        max(@maxMemoryUsed / 1000 / 1000) as maxMemoryUsedMB,
        provisonedMemoryMB - maxMemoryUsedMB as overProvisionedMB
```

建立延遲報告。

```
filter @type = "REPORT" |
    stats avg(@duration), max(@duration), min(@duration) by bin(5m)
```

搜尋緩慢的函數調用，並消除重試或用戶端程式碼可能產生的重複請求。在此查詢中，**@duration** 以毫秒為單位。

```
fields @timestamp, @requestId, @message, @logStream
| filter @type = "REPORT" and @duration > 1000
| sort @timestamp desc
| dedup @requestId
```

```
| limit 20
```

Amazon VPC 流程日誌的查詢

尋找主機之間的前 15 個封包傳輸：

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

尋找特定子網路上主機的前 15 個位元組傳輸。

```
filter isIpv4InSubnet(srcAddr, "192.0.2.0/24")
    | stats sum(bytes) as bytesTransferred by dstAddr
    | sort bytesTransferred desc
    | limit 15
```

尋找使用 UDP 做為資料傳輸協定的 IP 地址。

```
filter protocol=17 | stats count(*) by srcAddr
```

尋找在擷取時段略過流程記錄的 IP 地址。

```
filter logStatus="SKIPDATA"
    | stats count(*) by bin(1h) as t
    | sort t
```

尋找每個連線的單一記錄，以協助疑難排解網路連線問題。

```
fields @timestamp, srcAddr, dstAddr, srcPort, dstPort, protocol, bytes
| filter logStream = 'vpc-flow-logs' and interfaceId = 'eni-0123456789abcdef0'
| sort @timestamp desc
| dedup srcAddr, dstAddr, srcPort, dstPort, protocol
| limit 20
```

Route 53 日誌的查詢

依查詢類型尋找每小時的記錄分佈。

```
stats count(*) by queryType, bin(1h)
```

尋找請求數最高的前 10 個 DNS 解析程式。

```
stats count(*) as numRequests by resolverIp
    | sort numRequests desc
    | limit 10
```

依網域和子網域尋找伺服器無法完成 DNS 請求的記錄數。

```
filter responseCode="SERVFAIL" | stats count(*) by queryName
```

CloudTrail 日誌的查詢

尋找每個服務、事件類型和 AWS 區域的日誌項目數量。

```
stats count(*) by eventSource, eventName, awsRegion
```

尋找在指定 AWS 區域中啟動或停止的 Amazon EC2 主機。

```
filter (eventName="StartInstances" or eventName="StopInstances") and awsRegion="us-
east-2"
```

尋找新建立的 IAM 使用者 AWS 的區域、使用者名稱和 ARNs。

```
filter eventName="CreateUser"
    | fields awsRegion, requestParameters.userName, responseElements.user.arn
```

尋找叫用 API **UpdateTrail** 時發生例外狀況的記錄數。

```
filter eventName="UpdateTrail" and ispresent(errorCode)
    | stats count(*) by errorCode, errorMessage
```

尋找使用 TLS 1.0 或 1.1 的日誌條目

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
```

```
| stats count(*) as numOutdatedTlsCalls by userIdentity.accountId, recipientAccountId,
 eventSource, eventName, awsRegion, tlsDetails.tlsVersion, tlsDetails.cipherSuite,
 userAgent
| sort eventSource, eventName, awsRegion, tlsDetails.tlsVersion
```

尋找使用 TLS 1.0 或 1.1 版本之每項服務的呼叫次數

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
| stats count(*) as numOutdatedTlsCalls by eventSource
| sort numOutdatedTlsCalls desc
```

的查詢 Amazon API Gateway

找出最後 10 個 4XX 錯誤

```
fields @timestamp, status, ip, path, httpMethod
| filter status>=400 and status<=499
| sort @timestamp desc
| limit 10
```

識別 Amazon API Gateway 存取日誌群組中執行時間最長的 10 個 Amazon API Gateway 請求

```
fields @timestamp, status, ip, path, httpMethod, responseLatency
| sort responseLatency desc
| limit 10
```

傳回 Amazon API Gateway 存取日誌群組中最常用的 API 路徑清單

```
stats count(*) as requestCount by path
| sort requestCount desc
| limit 10
```

為您的 Amazon API Gateway 存取日誌群組建立整合延遲報告

```
filter status=200
| stats avg(integrationLatency), max(integrationLatency),
min(integrationLatency) by bin(1m)
```

NAT 閘道的查詢

如果您在 AWS 帳單中發現高於正常成本，您可以使用 CloudWatch Logs Insights 來尋找主要參與者。如需下列查詢命令的詳細資訊，請參閱 AWS 進階支援頁面中的如何透過 VPC 中的 NAT 閘道尋找流量的主要參與者？。

> **Note**
>
> 在以下查詢命令中，將 "x.x.x.x" 取代為 NAT 閘道的私有 IP，並將 "y.y" 替換為 VPC CIDR 範圍的前兩個八位元組。

查看透過 NAT 閘道傳送最多流量的執行個體。

```
filter (dstAddr like 'x.x.x.x' and srcAddr like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

確定進出 NAT 閘道中執行個體的流量。

```
filter (dstAddr like 'x.x.x.x' and srcAddr like 'y.y.') or (srcAddr like 'xxx.xx.xx.xx'
 and dstAddr like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

確定 VPC 中的執行個體在上傳和下載時，最經常與之通訊的網際網路目的地。

對於上傳

```
filter (srcAddr like 'x.x.x.x' and dstAddr not like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

對於下載

```
filter (dstAddr like 'x.x.x.x' and srcAddr not like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
```

```
| sort bytesTransferred desc
| limit 10
```

Apache 伺服器日誌的查詢

您可以使用 CloudWatch Logs Insights 來查詢 Apache 伺服器日誌。如需下列查詢的詳細資訊，請參閱 AWS Cloud Operations & Migrations 部落格中的使用 CloudWatch Logs Insights 簡化 Apache 伺服器日誌。

查看最相關的欄位，以在應用程式的 /admin 路徑中檢閱存取日誌並檢查流量。

```
fields @timestamp, remoteIP, request, status, filename| sort @timestamp desc
| filter filename="/var/www/html/admin"
| limit 20
```

查找以狀態碼 "200" (成功) 存取主頁面的不重複 GET 請求次數。

```
fields @timestamp, remoteIP, method, status
| filter status="200" and referrer= http://34.250.27.141/ and method= "GET"
| stats count_distinct(remoteIP) as UniqueVisits
| limit 10
```

查找 Apache 服務重新啟動的次數。

```
fields @timestamp, function, process, message
| filter message like "resuming normal operations"
| sort @timestamp desc
| limit 20
```

Amazon EventBridge 的查詢

取得按事件詳細資訊類型分組的 EventBridge 事件數

```
fields @timestamp, @message
| stats count(*) as numberOfEvents by `detail-type`
| sort numberOfEvents desc
```

剖析命令的範例

使用 glob 運算式，從日誌欄位 **@message** 中擷取欄位 **@user**、**@method** 和 **@latency**，並傳回 **@method** 和 **@user** 各種不重複組合的平均延遲。

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

使用規則運算式，從日誌欄位 **@message** 中擷取欄位 **@user2**、**@method2** 和 **@latency2**，並傳回 **@method2** 和 **@user2** 各種不重複組合的平均延遲。

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

擷取欄位 **loggingTime**、**loggingType** 和 **loggingMessage**，並篩選包含 **ERROR** 或 **INFO** 字串的日誌事件，然後針對包含 **ERROR** 字串的事件，僅顯示 **loggingMessage** 和 **loggingType** 欄位。

```
FIELDS @message
    | PARSE @message "* [*] *" as loggingTime, loggingType, loggingMessage
    | FILTER loggingType IN ["ERROR", "INFO"]
    | DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

比較（差異）與先前的時間範圍

您可以使用 CloudWatch Logs Insights 搭配 Logs Insights QL 來比較日誌事件中隨時間的變化。您可以將最近時間範圍擷取的日誌事件與上一個時段的日誌進行比較。或者，您可以與類似的過去時段進行比較。這可協助您找出日誌中的錯誤是最近推出的還是已經發生，並可協助您尋找其他趨勢。

比較查詢只會傳回結果中的模式，不會傳回原始日誌事件。傳回的模式可協助您快速查看日誌事件隨時間變化的趨勢和變化。執行比較查詢並取得模式結果後，您可以查看感興趣的模式的範例原始日誌事件。如需日誌模式的詳細資訊，請參閱 模式分析。

當您執行比較查詢時，會根據兩個不同的時段來分析查詢：您選取的原始查詢期間，以及比較期間。比較期間的長度一律與原始查詢期間相同。比較的預設時間間隔如下。

- 上一個期間 — 與查詢期間之前的期間進行比較。
- 前一天 — 與查詢期間前一天的期間進行比較。
- 上週 — 與查詢期間前一週的期間進行比較。
- 上個月 — 與查詢期間之前的一個月期間進行比較。

> **Note**
>
> 使用比較的查詢會產生類似於在合併時間範圍內執行單一 CloudWatch Logs Insights 查詢的費用。如需詳細資訊，請參閱 Amazon CloudWatch 定價。

執行比較查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌，然後選擇 Logs Insights。

   預設查詢會出現在查詢方塊中。
3. 確認已選取 Logs Insights QL 標籤。
4. 保留預設查詢或輸入不同的查詢。
5. 在選取日誌群組 （ 選取日誌群組 ） 下拉式清單中，選擇要查詢的一或多個日誌群組。
6. (選用) 使用時間間隔選擇器，選取您要查詢的時間段。預設查詢適用於前一小時的日誌資料。
7. 根據時間範圍選擇器，選擇比較。然後選擇您要與之比較原始日誌的先前時段，然後選擇套用。
8. 選擇 Run query (執行查詢)。

   為了讓查詢從比較期間擷取資料， `diff`命令會附加到您的查詢。
9. 選擇模式索引標籤以查看結果。

   資料表會顯示下列資訊：

   - 每個模式，模式的可變部分以動態字符符號 取代<*string-number*>。*##*是字符所代表資料類型的描述。此*##*顯示與其他動態字符相比，此字符在模式中的顯示位置。如需詳細資訊，請參閱模式分析。
   - 事件計數是具有該模式的日誌事件數量，在原始、較目前的時段內。
   - 差異事件計數是目前期間內相符日誌事件的數量與比較期間之間的差異。正數不同表示目前期間內有更多這類事件。
   - 差異描述簡短總結了目前期間和比較期間之間該模式的變更。
   - 嚴重性類型是根據日誌事件中找到的字詞，例如 `FATAL`、 `ERROR`和 ，以此模式記錄事件的可能嚴重性`WARN`。
10. 若要進一步檢查清單中的其中一個模式，請選擇檢查欄中其中一個模式的圖示。

    模式檢查窗格隨即出現，並顯示下列項目：

- 模式。在模式中選取權杖，以分析該權杖的值。
- 顯示查詢時間範圍內模式出現次數的長條圖。這可協助您識別有趣的趨勢，例如模式的出現突然增加。
- 日誌範例索引標籤會顯示一些符合所選模式的日誌事件。
- 如果您已選取動態權杖，權杖值索引標籤會顯示所選動態權杖的值。

> **Note**
>
> 每個字符最多擷取 10 個字符值。字符計數可能不精確。CloudWatch Logs 使用機率計數器來產生字符計數，而不是絕對值。

- 相關模式索引標籤會顯示與您檢查的模式幾乎同時經常發生的其他模式。例如，如果ERROR訊息的模式通常伴隨另一個標記為 的日誌事件INFO和其他詳細資訊，則此模式會顯示在此處。

在圖表中視覺化日誌資料

您可以使用長條圖、折線圖和堆疊區域圖等視覺效果，更有效率地識別日誌資料中的模式。CloudWatch Logs Insights 可以為使用 `stats` 函數和一或多個彙總函數的查詢產生視覺效果。如需詳細資訊，請參閱 stats。

OpenSearch 管道處理語言 (PPL)

本節包含使用 OpenSearch PPL 查詢 CloudWatch Logs 的基本簡介。您可以使用 PPL 擷取、查詢和分析資料，以便更輕鬆地了解和編寫複雜的查詢。其語法是以 Unix 管道為基礎，並啟用命令鏈結來轉換和處理資料。透過 PPL，您可以篩選和彙總資料，並使用一組豐富的數學、字串、日期、條件式和其他函數進行分析。

在 PPL 查詢SOURCE中包含 是指定日誌群組欄位索引的實用方式，以及當您使用 AWS CLI 或 API 建立查詢時要包含在查詢中的資料來源。SOURCE 命令僅在 AWS CLI 和 API 中受支援，在 CloudWatch 主控台中不受支援。當您使用 CloudWatch 主控台啟動查詢時，您可以使用主控台界面來指定日誌群組和資料來源名稱和類型。

`aws:fieldIndex` 使用 僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。系統會根據 `filterIndex`命令中指定的欄位，自動選取相關的日誌群組。這可藉由略過沒有任何包含查詢中指定欄位之日誌事件的日誌群組，以及僅掃描符合此欄位索引查詢中指定值的日誌群組，來減少掃描的磁碟區。使用 `aws:fieldIndex`指定欄位名稱，以及來源命令中的欄位名稱和值，

以僅查詢包含指定欄位和值的索引資料。如需詳細資訊，請參閱建立欄位索引以改善查詢效能並減少掃描磁碟區

您可以使用 OpenSearch PPL 來查詢標準日誌類別中的日誌群組。

> **Note**
>
> 如需 CloudWatch Logs 中支援的所有 OpenSearch PPL 查詢命令的相關資訊，以及語法和限制的詳細資訊，請參閱OpenSearch Service 開發人員指南》中的支援的 PPL 命令。
> 如需有關您可以使用之其他查詢語言的資訊，請參閱 CloudWatch Logs Insights、OpenSearch Service SQL 和 CloudWatch Metrics Insights

命令或函數	查詢範例	Description
fields	`fields field1, field2`	顯示一組需要投影的欄位。
join	`LEFT JOIN left=l, right=r on l.id = r.id` `` `join_right_lg` `` `| fields l.field_1, r.field_2`	將兩個資料集聯結在一起。
where	`where field1="success" | where field2 != "i-023fe0a90929d8822" | fields field3, field4, field5,field6 | head 1000`	根據您指定的條件篩選資料。
aws：fieldIndex	`source = [` `` `aws:fieldIndex` `` `="region", ` `` `region` `` `= "us-west-2"] | where status = 200 | head 10`	僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。
統計資料	`stats count(), count(field1), min(field 1), max(field1), avg(field1) by field2 | head 1000`	執行彙總和計算

命令或函數	查詢範例	Description
parse	`parse field1 ".*/(?<field2>[^/]+$)" | where field2 = "requestId" | fields field1, field2 | head 1000`	從字串擷取規則表達式 (regex) 模式，並顯示擷取的模式。擷取的模式可以進一步用來建立新的欄位或篩選資料。
sort	`stats count(), count(field1), min(field1) as field1Alias, max(`field1`), avg(`field1`) by field2 | sort -field1Alias | head 1000`	依欄位名稱排序顯示的結果。使用 sort - FieldName 以遞減順序排序。
評估	`eval field2 = field1 * 2 | fields field1, field2 | head 20`	修改或處理欄位的值，並將其存放在不同的欄位中。這有助於以數學方式修改資料欄、將字串函數套用至資料欄，或將日期函數套用至資料欄。
重新命名	`rename field2 as field1 | fields field1;`	重新命名搜尋結果中的一個或多個欄位。
head	`fields `@message` | head 20`	將顯示的查詢結果限制為前 N 列。
top	`top 2 field1 by field2`	尋找欄位最常見的值。

命令或函數	查詢範例	Description
dedup	`dedup field1 | fields field1, field2, field3`	根據您指定的欄位移除重複的項目。
很少見	`rare field1 by field2`	尋找欄位清單中所有欄位頻率最低的值。
subquery	`where field_1 IN [ search source= `subquery_lg` | fields field_2 ] | fields id, field_1`	在您的 PPL 陳述式中執行複雜的巢狀查詢。
趨勢線	`trendline sma(2, field1) as field1Alias`	計算欄位的移動平均值。
eventStats	`eventstats sum(field1) by field2`	使用計算的摘要統計資料來豐富您的事件資料。它會分析事件中的指定欄位、運算各種統計指標，然後將這些結果做為新欄位附加到每個原始事件。
expand	`eval tags_array_string = json_extr act(`@message`, '$.tags')\| eval tags_array = json_array(json_ex tract(tags_string, '$[0]'), json_extr act(tags_string, '$[1]'))| expand tags_array as color_tags`	將包含多個值的欄位細分為不同的資料列，為指定欄位中的每個值建立新的資料列。

命令或函數	查詢範例	Description
fillnull	`fields `@timestamp`, error_code, status_code | fillnull using status_code = "UNKNOWN", error_code = "UNKNOWN"`	使用您提供的值填入 null 欄位。它可用於一個或多個欄位。
flatten	`eval metadata_struct = json_object('size', json_extract(metadata_string, '$.size'), 'color', json_extract(metadata_string, '$.color')) | flatten metadata_struct as (meta_size, meta_color)`	扁平化欄位。欄位必須屬於此類型：`struct<?,?>`或 `array<struct<?,?>>` 。
cidrmatch	`where cidrmatch(ip, '2003:db8::/32') | fields ip`	檢查指定的 IP 地址是否在指定的 CIDR 範圍內。
欄位摘要	`where field1 != 200 | fieldsummary includefields= field1 nulls=true`	計算每個欄位的基本統計資料（計數、相異計數、最小值、最大值、平均值、標準差和平均值）。
grok	`grok email '.+@%{HOSTNAME:host}' | fields email, host`	剖析具有 grok 模式的文字欄位，並將結果附加到搜尋結果。

命令或函數	查詢範例	Description
字串函數	`eval field1Len = LENGTH(field1) | fields field1Len`	PPL 中的內建函數，可以操作和轉換 PPL 查詢中的字串和文字資料。例如，轉換案例、合併字串、擷取部分和清理文字。
日期時間函數	`eval newDate = ADDDATE(DATE('2020-08-26'), 1) | fields newDate`	用於處理和轉換 PPL 查詢中日期和時間戳記資料的內建函數。例如，date_add、date_format、datediff、date-sub、timestampadd、timestampdiff、current_timezone、utc_timestamp 和 current_date。
條件函數	`eval field2 = isnull(field1) | fields field2, field1, field3`	檢查特定欄位條件，並依條件評估表達式的內建函數。例如，如果 field1 為 null，則傳回 field2。

命令或函數	查詢範例	Description
數學函數	`eval field2 = ACOS(field1) | fields field1`	用於在 PPL 查詢中執行數學計算和轉換的內建函數。例如，abs（絕對值）、圓（圓數）、sqrt（平方根）、pow（功率計算）和 ceil（四捨五入至最接近的整數）。
CryptoGraphic 函數	`eval crypto = MD5(field)| head 1000`	計算指定欄位的雜湊
JSON 函數	`eval valid_json = json('[1,2,3,{"f1":1,"f2":[5,6]},4]') | fields valid_json`	用於處理 JSON 的內建函數，包括陣列、擷取和驗證。例如，json_object、json_array、to_json_string、json_array_length、json_extract、json_keys 和 json_valid。

查詢範圍

當您使用 AWS CLI 或 API 建立查詢時，在查詢中包含 SOURCE 是指定要包含在查詢中的日誌群組的實用方式。SOURCE 命令僅支援 AWS CLI 和 API，不支援 CloudWatch 主控台。當您使用 CloudWatch 主控台啟動查詢時，您可以使用主控台界面來指定日誌群組和資料來源名稱和類型。

PPL 的來源命令現在支援多種指定方式：

1. 日誌群組
2. 欄位索引 - 新
3. 資料來源和類型 - 新

日誌群組

當客戶知道需要搜尋哪個確切的日誌群組（日誌群組）時，可以使用日誌群組來源選擇

```
source = [lg:`/aws/lambda/my-function`] | where status = 200 | head 10
```

欄位索引

當篩選條件鎖定已編製索引的欄位時，將結果限制為僅編製索引的資料，以欄位索引為基礎的來源選擇可減少查詢的資料量。系統會根據 `filterIndex`命令中指定的欄位，自動選取相關的日誌群組。如需欄位索引及其建立方式的詳細資訊，請參閱建立欄位索引以改善查詢效能並降低掃描磁碟區。

`aws:fieldIndex` 使用 僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。對於在此欄位上編製索引的這些日誌群組，它會略過沒有包含索引欄位查詢中指定欄位的任何日誌事件的日誌群組，以進一步最佳化查詢。它透過嘗試僅掃描這些日誌群組中符合此欄位索引查詢中指定值的日誌事件，進一步減少掃描的磁碟區。如需欄位索引及其建立方式的詳細資訊，請參閱建立欄位索引以改善查詢效能並降低掃描磁碟區。

在 PPL 中， `aws:fieldIndex` 用於指定哪些索引鍵值對應被視為索引。語法如下

```
source = [`aws:fieldIndex`="region", `region` = "us-west-2"] | where status = 200 |
 head 10
```

其中，

1. `` `aws:fieldIndex`="region" `` 將區域識別為欄位索引。
   a. 注意：而不是 = 客戶可以使用 IN 來指定多個索引（範例如下）

2. `region`="us-west-2" 識別要套用的篩選條件

   a. 注意：而不是 = 客戶可以使用 IN 來指定多個值 （ 範例如下 ）

客戶可以指定多個 fieldIndexes，如下所示

```
source = [`aws:fieldIndex` IN ("status", "region"), `status` = 200, `region` IN ("us-
west-2", "us-east-1")] | head 10
```

資料來源和類型

當客戶知道需要查詢哪些確切資料來源時，可以使用資料來源和類型型來源選擇。此查詢會透過一或多個包含指定資料來源和類型的日誌群組執行。

```
source = [ds:`data_source.type`] | where status = 200 | head 10
```

資料來源查詢支援的 PPL

若要支援在 PPL 中查詢資料來源的使用案例，您可以使用動態來源選取器子句。使用此語法，您可以在搜尋命令中指定資料來源，以查詢資料來源。您最多可以指定 10 個資料來源。

語法

```
source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`, ...ds:`DataSourcen.Typen`]
```

查詢範例

```
search source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`] | fields field1, field2
```

合併的範例

客戶可以按任何順序指定所有來源選取運算子，結果會是所有套用條件的交集。

例如，/aws/lambda/my-function-1 可能包含多個資料來源和類型，包括各種索引，當執行下列查詢時，傳回的結果只會有來源和類型 DataSource1.Type1 的事件，並符合 'status' = 200 的條件。

```
search source=[
    ds:`DataSource1.Type1`,
    lg:`/aws/lambda/my-function-1`,
    `aws:fieldIndex` IN ("status"), `status` = 200
]
```

限制

當您使用 OpenSearch PPL 在 CloudWatch Logs Insights 中查詢時，適用下列限制。

- 您無法搭配資料來源查詢使用聯結或子查詢命令。

OpenSearch 結構化查詢語言 (SQL)

本節包含使用 OpenSearch SQL 查詢 CloudWatch Logs 的基本簡介。如果您習慣使用關聯式資料庫，它會提供熟悉的選項。OpenSearch SQL 提供 SQL 功能的子集，使其成為執行臨機操作查詢和資料分析任務的理想選擇。透過 OpenSearch SQL，您可以使用 SELECT、 FROM、WHERE、GROUP BY、HAVING 等命令，以及各種其他 SQL 命令和函數。您可以跨日誌群組執行 JOINs、使用子查詢跨日誌群組關聯資料，並使用一組豐富的 JSON、數學、字串、條件式和其他 SQL 函數，對日誌和安全資料執行強大的分析。

`filterIndex` 使用 僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。略過沒有任何包含查詢中指定欄位之日誌事件的日誌群組，並僅掃描符合此欄位索引查詢中指定值的日誌群組，以減少掃描的磁碟區。使用 `filterIndex`指定欄位名稱，以及欄位名稱和值，以僅查詢包含指定欄位和值的索引資料。

您可以使用 OpenSearch SQL 來查詢標準日誌類別中的日誌群組。SQL 也支援使用資料來源名稱和資料來源類型進行查詢。

> **Note**
>
> 下表列出 CloudWatch Logs 中支援的 SQL 命令和函數 如需所有 OpenSearch SQL 命令的資訊，包括語法，請參閱 OpenSearch Service 開發人員指南中的支援的 SQL 命令。
> 如需有關您可以使用的其他查詢語言的資訊，請參閱CloudWatch Logs Insights、OpenSearch Service PPL 和 CloudWatch。 Metrics Insights

支援的 SQL 命令

> **Note**
>
> 在範例查詢欄中，根據您查詢的資料來源*<logGroup>*，視需要取代 。

命令或函數	查詢範例	Description
SELECT	`SELECT `@message`, Operation FROM `LogGroupA``	顯示投影值。
FROM	`SELECT `@message`, Operation FROM `LogGroupA``	內建子句（指定要從中擷取資料的來源資料表或檢視），支援各種類型的聯結和子查詢。
WHERE	`SELECT * FROM `LogGroupA` WHERE Operation = 'x'`	根據提供的欄位條件篩選日誌事件。
filterIndex	`SELECT * FROM `filterIndex('region' = 'us-east-1')` WHERE status = 200 LIMIT 10;`	僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。
GROUP BY	`SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream``	根據類別對日誌事件進行分組，並根據統計資料尋找平均值。
HAVING	`SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream` HAVING log_count > 100`	根據分組條件篩選結果。
ORDER BY	`SELECT * FROM `LogGroupA` ORDER BY `@timestamp` DESC`	根據 ORDER BY 子句中的欄位排序結果。您可以依遞減或遞增順序排序。

命令或函數	查詢範例	Description
JOIN	`SELECT A.`@message`, B.`@timestamp`FROM `LogGroupA` as A INNER JOIN `LogGroupB` as B ON A.`requestId` = B.`requestId``	根據常見欄位聯結兩個資料表的結果。必須指定內部 JOIN 或左側外部聯結
LIMIT	`Select * from `LogGroupA` limit 10`	將顯示的查詢結果限制為前 N 列。
字串函數	`SELECT upper(Operation) , lower(Operation), Operation FROM `LogGroupA``	SQL 中的內建函數，可以操作和轉換 SQL 查詢中的字串和文字資料。例如，轉換案例、合併字串、擷取部分和清理文字。
日期函數	`SELECT current_date() as today, date_add(current_date(), 30) as thirty_days_later, last_day(current_date()) as month_end FROM `LogGroupA``	用於處理和轉換 SQL 查詢中日期和時間戳記資料的內建函數。例如，date_add、date_format、datediff 和 current_date。
條件函數	`SELECT Operation, IF(Error > 0, 'High', 'Low') as error_category FROM `LogGroupA`;`	內建函數，可根據指定的條件執行動作，或以條件方式評估表達式。例如，CASE 和 IF。

命令或函數	查詢範例	Description
彙總函數	`SELECT AVG(bytes) as bytesWritten FROM `LogGroupA``	對多個資料列執行計算以產生單一摘要值的內建函數。例如，SUM、COUNT、AVG、MAX 和 MIN。
JSON 函數	`SELECT get_json_object(json_column, '$.name') as name FROM `LogGroupA``	用於在 SQL 查詢中剖析、擷取、修改和查詢 JSON 格式資料的內建函數（例如，from_json、to_json、get_json_object、json_tuple），允許在資料集中操作 JSON 結構。
陣列函數	`SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;`	用於在 SQL 查詢中使用陣列類型資料欄的內建函數，允許存取、修改和分析陣列資料（例如，大小、爆炸、Array_contains) 等操作。

命令或函數	查詢範例	Description
範圍函數	`SELECT field1, field2, RANK() OVER (ORDER BY field2 DESC) as field2Rank FROM `LogGroupA`;`	內建函數，可在與目前資料列（視窗）相關的指定資料列中執行計算，啟用排名、執行總計和移動平均值等操作。例如，ROW_NUMBER、RANK、LAG 和 LEAD
轉換函數	`SELECT CAST('123' AS INT) as converted_number, CAST(123 AS STRING) as converted_string FROM `LogGroupA``	用於在 SQL 查詢中將資料從一種類型轉換為另一種類型的內建函數，啟用資料類型轉換和格式轉換。例如，CAST、TO_DATE、TO_TIMESTAMP 和 BINARY。
述詞函數	`SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;`	根據指定的條件或模式，評估條件並傳回布林值 (true/false) 的內建函數。例如，IN、LIKE、BETWEEN、IS NULL 和 EXISTS。

命令或函數	查詢範例	Description
選取多個日誌群組	`SELECT lg1.field1, lg1.field2 from `logGroups(logGroupIdentifier: ['LogGroup1', 'LogGroup2'])` as lg1 where lg1.field3= "Success"`	可讓您在 SELECT 陳述式中指定多個日誌群組
選取多個資料來源	`SELECT ds1.field1, ds1.field2 from `dataSource(['DataSource1', 'DataSource2'])` as ds1 where ds1.field3= "Success"`	可讓您在 SELECT 陳述式中指定多個資料來源

multi-log-group查詢支援的 SQL

若要支援在 SQL 中查詢多個日誌群組的使用案例，您可以使用 `logGroups`命令。使用此語法，您可以在 FROM 命令中指定多個日誌群組來查詢它們。

語法：

```
`logGroups(
    logGroupIdentifier: ['LogGroup1','LogGroup2', ...'LogGroupn']
)
```

在此語法中，您可以在 `logGroupIdentifier` 參數中指定最多 50 個日誌群組。若要參考監控帳戶中的日誌群組，請使用 ARNs 而非`LogGroup`名稱。

查詢範例：

```
SELECT LG1.Column1, LG1.Column2 from `logGroups(
    logGroupIdentifier: ['LogGroup1', 'LogGroup2']
)` as LG1 WHERE LG1.Column1 = 'ABC'
```

查詢 CloudWatch Logs 時，不支援在`FROM`陳述式之後涉及多個日誌群組的下列語法。

```
SELECT Column1, Column2 FROM 'LogGroup1', 'LogGroup2', ...'LogGroupn'
WHERE Column1 = 'ABC'
```

資料來源查詢支援的 SQL

若要支援在 SQL 中查詢資料來源的使用案例，您可以使用 dataSource 命令。使用此語法，您可以在 FROM命令中指定資料來源，以查詢資料來源。您最多可以指定 10 個資料來源。

語法

```
`dataSource(
    ['DataSource1', 'DataSource2', ...'DataSourcen']
)`
```

查詢範例

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(
    ['DataSource1', 'DataSource2']
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

查詢範圍

在 AWS CLI 和 API 中，您可以使用日誌群組、資料來源和類型，以及欄位索引來指定要查詢的日誌。

日誌群組

當客戶知道需要搜尋哪個確切日誌群組 （ 日誌群組 ） 時，可以使用日誌群組來源選擇

```
SELECT * FROM `logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])`;
```

資料來源和類型

客戶可以使用資料來源名稱和資料來源類型來查詢其日誌。

當客戶知道需要查詢哪些確切資料來源時，可以使用資料來源和類型型來源選擇。此查詢會透過一或多個包含指定資料來源和類型的日誌群組執行。

若要支援在 SQL 中查詢資料來源的使用案例，您可以使用 dataSource 命令。使用此語法，您可以在 FROM 命令中指定資料來源，以查詢資料來源。您最多可以指定 10 個資料來源。

語法：

```
`dataSource(
    ['DataSource1.Type1', 'DataSource2.Type2', ...'DataSourcen.Typen']
)`
```

查詢範例：

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(
    ['DataSource1.Type1', 'DataSource2.Type2']
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

如需依資料來源查詢的詳細資訊，請參閱 使用面向來分組和探索日誌。

合併的範例

客戶可以在反引號內以任何順序指定所有來源選取運算子，結果會根據所有套用條件的交集。

例如，/aws/lambda/my-function-1 可能包含多個資料來源和類型，包括各種索引，當執行下列查詢時，傳回的結果只會有來源和類型 DataSource1.Type1 的事件，並符合 'status' = 200 的條件。

```
SELECT * FROM `
   logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])
   filterIndex('status' = 200)
   dataSource(['DataSource1.Type1'])
`;
```

欄位索引

當您篩選目標索引欄位時，欄位索引型來源選擇會自動識別相關日誌群組，以減少掃描磁碟區和查詢執行時間。

`filterIndex` 使用 僅傳回索引資料，方法是強制查詢僅掃描您在查詢中指定的欄位上編製索引的日誌群組。對於在此欄位上編製索引的這些日誌群組，它會略過沒有包含索引欄位查詢中指定欄位之任何日誌事件的日誌群組，以進一步最佳化查詢。它透過嘗試僅掃描這些日誌群組中符合此欄位索引查詢中指定值的日誌事件，進一步減少掃描的磁碟區。如需欄位索引以及如何建立它們的詳細資訊，請參閱建立欄位索引以改善查詢效能並減少掃描磁碟區。

在 SQL 中，filterIndex 用於指定哪些索引鍵值對應被視為索引。語法如下

```
SELECT * FROM `filterIndex('region' = 'us-east-1')`;
```

其中，

1. filterIndex(...) 指定將其中的索引鍵值視為欄位索引。每個索引鍵值對以逗號分隔 （ 範例如下 ）
2. 'region' = 'us-east-1' 指定要套用的實際條件
   a. 注意：而不是 = 客戶可以使用 IN 來指定多個值 （ 範例如下 ）

使用多個 filterIndex 會使用 "AND" 結合條件。在此範例中，會查詢 us-east-1 或 us-west-2 中的日誌比對狀態=200 和區域。

```
SELECT * FROM `filterIndex('status' = 200, 'region' IN ['us-east-1', 'us-west-2'])`;
```

限制

當您使用 OpenSearch SQL 在 CloudWatch Logs Insights 中查詢時，適用下列限制。

- 您只能在 SELECT 陳述式中包含一個 JOIN。
- 您無法搭配資料來源查詢使用 JOIN 或子查詢。
- 僅支援一個層級的巢狀子查詢。
- 不支援以分號 (;) 分隔的多個陳述式查詢。
- 不支援包含相同但僅在 欄位 （ 例如 field1 和 FIELD1) 時才不同的欄位名稱的查詢。

  例如，不支援下列查詢：

  ```
  Select AWSAccountId, AwsAccountId from LogGroup
  ```

  不過，由於兩個日誌群組中的欄位名稱 (@logStream) 相同，因此支援下列查詢：

  ```
  Select a.`@logStream`, b.`@logStream` from Table A INNER Join Table B on a.id = b.id
  ```

- 函數和表達式必須在欄位名稱上操作，並成為 SELECT 陳述式的一部分，其中包含 FROM 子句中指定的日誌群組。

  例如，不支援此查詢：

  ```
  SELECT cos(10) FROM LogGroup
  ```

  支援此查詢：

  ```
  SELECT cos(field1) FROM LogGroup
  ```

- 使用 SQL 或 PPL 命令時，請將特定欄位括在反引號中，以成功查詢它們。具有特殊字元 （ 非字母和非數字 ） 的欄位需要反引號。例如，將 @message、 Operation.Export和 括在反引號Test::Field中。您不需要在反引號中以純字母名稱括住欄位。

  具有簡單欄位的範例查詢：

```
SELECT SessionToken, Operation, StartTime  FROM `LogGroup-A`
LIMIT 1000;
```

附加反引號的類似查詢：

```
SELECT `@SessionToken`, `@Operation`, `@StartTime`  FROM `LogGroup-A` LIMIT 1000;
```

使用自然語言來產生和更新 CloudWatch Logs Insights 查詢

CloudWatch Logs 支援自然語言查詢功能，可協助您產生和更新 CloudWatch Logs Insights、OpenSearch Service PPL、OpenSearch Service SQL 和 CloudWatch Metrics Insights的查詢。

使用此功能，您可以詢問或描述以純英文尋找的 CloudWatch Logs 資料。自然語言功能會根據您輸入的提示產生查詢，並逐行說明查詢的運作方式。也可以更新查詢以進一步調查您的資料。

根據您的環境，您可以輸入提示，例如「傳輸位元組的前 100 個來源 IP 地址是什麼？」 和「尋找 10 個最慢的 Lambda 函數請求。」

> **Note**
>
> 自然語言查詢功能是區域性服務。對於某些區域，此功能會對美國區域進行跨區域呼叫，以處理查詢提示。如需詳細資訊，請參閱 Amazon CloudWatch 擴展區域對自然語言查詢結果摘要和查詢產生 的支援。

若要使用此功能產生 CloudWatch Logs Insights 查詢，請開啟 CloudWatch Logs Insights 查詢編輯器，選取您要查詢的日誌群組，然後選擇產生查詢。

> **Important**
>
> 若要使用自然語言查詢功能，您必須使用 CloudWatchLogsFullAccess、CloudWatchLogsReadOnlyAccess、 AdministratorAccess 或 ReadOnlyAccess IAM 政策登入，或具有 `cloudwatch:GenerateQuery`許可。

查詢範例

本節中的範例說明如何使用自然語言功能產生及更新查詢。

> **Note**
>
> 如需 CloudWatch Logs Insights 查詢編輯器和語法的詳細資訊，請參閱 CloudWatch Logs Insights 查詢語法。

範例：產生自然語言查詢

若要使用自然語言產生查詢，請輸入提示並選擇產生新查詢。這些範例顯示執行基本搜尋的查詢。

提示詞

以下是提示範例，指示 搜尋 10 個最慢 Lambda 函數叫用的功能。

```
Find the 10 slowest requests
```

Query

以下是使用 CloudWatch Logs Insights 查詢語言的查詢，自然語言功能會根據提示產生。請注意提示在查詢前出現在註解中的方式。查詢之後，您可以閱讀描述查詢運作方式的說明。

```
# Find the 10 slowest requests
fields @timestamp, @message, @duration
| sort @duration desc
| limit 10
# This query retrieves the timestamp, message and duration fields from the logs and
 sorts them in descending order by duration to find the 10 slowest requests.
```

> **Note**
>
> 若要關閉提示的外觀以及查詢運作方式的說明，請使用編輯器中的齒輪圖示。

提示詞

若要產生 OpenSearch SQL 查詢，請選取 OpenSearch SQL 索引標籤，然後開啟查詢產生器提示方塊以輸入您的自然語言提示。以下是使用自然語言功能產生 OpenSearch SQL 查詢的提示範例。

```
Give me the number of errors and exceptions per hour
```

Query

以下是該提示產生的 SQL 查詢，您可以使用它來尋找每小時彙總的錯誤和例外狀況數量：

```
SELECT DATE_FORMAT(`@timestamp`, 'yyyy-MM-dd HH') AS hour,
       COUNT(*) AS error_count
FROM `/aws/lambda/CloudWatchOdysseyQueryGen`
WHERE `@message` LIKE '%error%'
  OR `@message` LIKE '%exception%'
GROUP BY DATE_FORMAT(`@timestamp`, 'yyyy-MM-dd HH')
ORDER BY hour
```

提示詞

若要產生 OpenSearch PPL 查詢，請選取 OpenSearch PPL 標籤，然後開啟查詢產生器提示方塊以輸入您的自然語言提示。以下是使用自然語言功能來產生 OpenSearch PPL 查詢的提示範例。

```
Give me all unique exception messages
```

Query

以下是該提示產生的 PPL 查詢，您可以使用它在日誌中尋找唯一的例外狀況訊息：

```

dedup @message
| fields @message

```

範例：更新自然語言查詢

可以透過編輯初始提示，然後選擇更新查詢來更新查詢。

更新提示

下列範例顯示先前提示的更新版本。而不是搜尋 10 個最慢 Lambda 函數調用的提示，此提示現在會指示 搜尋 20 個最慢 Lambda 函數調用的功能，並包含其他日誌事件的另一個資料欄。

```
Show top 20 slowest requests instead and display requestId as a column
```

更新查詢

以下是使用 CloudWatch Logs Insights 查詢語言更新查詢的範例。請注意更新後的提示在更新後的查詢前出現在註解中的方式。查詢之後，您可以閱讀描述原始查詢更新方式的說明。

```
# Show top 20 slowest requests instead and display requestId as a column
fields @timestamp, @message, @requestId, @duration
| sort @duration desc
| limit 20
# This query modifies the original query by replacing the @message field with the
 @requestId field and changing the limit from 10 to 20 to return the top 20 log events
 by duration instead of the top 10.
```

選擇不使用您的資料以改善服務

您提供用於訓練 AI 模型並產生相關查詢的自然語言提示資料僅用於提供和維護您的服務。此資料可能用於改善 CloudWatch Logs Insights 的品質。我們將您的信任和隱私以及內容安全性放在首位。如需詳細資訊，請參閱 AWS 服務條款和 AWS 負責任的 AI 政策。

透過建立 AI 服務退出政策，可選擇不將您的內容用於開發或改進自然語言查詢的品質。若要選擇退出所有 CloudWatch Logs AI 功能的資料收集，包括查詢產生功能，您必須為 CloudWatch Logs 建立選擇退出政策。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的 AI 服務退出政策。

支援的日誌和探索的欄位

CloudWatch Logs Insights 支援各種類型的日誌。對於傳送至 Amazon CloudWatch Logs 中標準類別日誌群組的每個日誌，CloudWatch Logs Insights 會自動產生五個系統欄位：

- `@message` 包含原始未分析的日誌事件。這相當於 InputLogevent 中的 `message` 欄位。
- `@timestamp` 含有日誌事件 `timestamp` 欄位中的事件時間戳記。這相當於 InputLogevent 中的 `timestamp` 欄位。
- `@ingestionTime` 含有 CloudWatch Logs 收到日誌事件的時間。
- `@logStream` 包含日誌事件新增到其中的日誌串流名稱。日誌串流透過產生日誌串流的相同程序對日誌進行分組。
- `@log` 是 *`account-id`*`:`*`log-group-name`* 形式的日誌群組識別碼。在查詢多個日誌群組時，這有助於識別特定事件所屬的日誌群組。

- `@entity` 包含與探索相關遙測功能的實體相關的平面化 JSON。

  例如，此 JSON 可以代表實體。

```
{
  "Entity": {
    "KeyAttributes": {
      "Type": "Service",
      "Name": "PetClinic"
    },
    "Attributes": {
      "PlatformType": "AWS::EC2",
      "EC2.InstanceId": "i-1234567890123"
    }
  }
}
```

  對於此實體，擷取的系統欄位如下：

```
@entity.KeyAttributes.Type = Service
@entity.KeyAttributes.Name = PetClinic
@entity.Attributes.PlatformType = AWS::EC2
@entity.Attributes.EC2.InstanceId = i-1234567890123
```

> **Note**
>
> 欄位探索僅支援標準日誌類別中的日誌群組。如需日誌類別的詳細資訊，請參閱 日誌類別。

CloudWatch Logs Insights 會在其產生的欄位開頭插入 @ 符號。

對於許多日誌類型，CloudWatch Logs 也會自動探索日誌包含的日誌欄位。下表顯示這些自動探索的欄位。

如果是 CloudWatch Logs Insights 不會自動探索欄位的其他日誌類型，您可以使用 `parse` 命令來擷取和建立擷取欄位，以用於該查詢中。如需詳細資訊，請參閱CloudWatch Logs Insights 語言查詢語法。

如果找到的日誌欄位以 @ 為名稱開頭，CloudWatch Logs Insights 顯示該欄位時會在開頭多加一個 @。例如，如果日誌欄位名稱是 `@example.com`，這個欄位名稱會顯示為 `@@example.com`。

> **Note**
>
> 除了 @message、 @timestamp或 之外@log，您可以為探索到的欄位建立欄位索引。如需欄位索引的詳細資訊，請參閱 建立欄位索引以改善查詢效能並減少掃描磁碟區。

日誌類型	探索的日誌欄位
Amazon VPC 流程日誌	@timestamp ，@logStream ，@message, accountId ，endTime, interfaceId ，logStatus ，startTime ，version, action, bytes, dstAddr, dstPort, packets, protocol, srcAddr, srcPort
Route 53 日誌	@timestamp ，@logStream ，@message, edgeLocation ，ednsClientSubnet ，hostZoneId ，protocol, queryName ，queryTimestamp ，queryType ，resolverIp ，responseCode ，version
Lambda 日誌	@timestamp ，@logStream ，@message, @requestId ，@duration, @billedDuration ，@type, @maxMemoryUsed ，@memorySize 如果 Lambda 日誌行包含 X-Ray 追蹤 ID，則也會包含以下欄位：@xrayTraceId 和 @xraySegmentId 。 CloudWatch Logs Insights 會自動探索 Lambda 日誌中的日誌欄位，但僅限於每個日誌事件中的第一個內嵌 JSON 片段。如果 Lambda 日誌事件包含多個 JSON 片段，您可以使用 **parse** 命令來剖析和擷取日誌欄位。如需詳細資訊，請參閱JSON 日誌中的欄位。
CloudTrail 日誌 JSON 格式的日誌	如需詳細資訊，請參閱JSON 日誌中的欄位。
其他日誌類型	@timestamp ，@ingestionTime ，@logStream ，@message, @log.

JSON 日誌中的欄位

藉由 CloudWatch Logs Insights，您可以使用點符號來表示 JSON 欄位。本節包含 JSON 事件範例和程式碼片段，示範如何使用點符號存取 JSON 欄位。

範例：JSON 事件

```
{
    "eventVersion": "1.0",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn: aws: iam: : 123456789012: user/Alice",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "accountId": "123456789012",
        "userName": "Alice"
    },
    "eventTime": "2014-03-06T21: 22: 54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "ec2-api-tools1.6.12.2",
    "requestParameters": {
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-abcde123"
                }
            ]
        }
    },
    "responseElements": {
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-abcde123",
                    "currentState": {
                        "code": 0,
                        "name": "pending"
                    },
                    "previousState": {
                        "code": 80,
```

```
                        "name": "stopped"
                    }
                }
            ]
        }
    }
}
```

範例 JSON 事件包含一個名為 `userIdentity` 的物件。`userIdentity` 包含名為 `type` 的欄位。若要使用點符號表示 `type` 的值，您可以使用 `userIdentity.type`。

範例 JSON 事件包含展平為巢狀欄位名稱和值清單的陣列。若要表示 `requestParameters.instancesSet` 中第一個項目 `instanceId` 的值，您可以使用 `requestParameters.instancesSet.items.0.instanceId`。放置在欄位 `instanceID` 前的數字 `0` 指的是欄位 `items` 的值的位置。下列範例包含一個程式碼片段，顯示如何存取 JSON 日誌事件中的巢狀 JSON 欄位。

範例：查詢

```
fields @timestamp, @message
| filter requestParameters.instancesSet.items.0.instanceId="i-abcde123"
| sort @timestamp desc
```

該程式碼片段顯示了一個查詢，該查詢使用帶有 `filter` 命令的點符號來存取巢狀 JSON 欄位 `instanceId` 的值。查詢會篩選出 `instanceId` 值等於 `"i-abcde123"` 的消息，並傳回包含指定值的所有日誌事件。

> **Note**
>
> CloudWatch Logs Insights 最多可從 JSON 日誌中擷取 200 個日誌事件欄位。針對未擷取的額外欄位，可以使用 `parse` 命令來擷取訊息欄位中原始未剖析日誌事件的欄位。如需有關 `parse` 命令的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的查詢語法。

建立欄位索引以改善查詢效能並減少掃描磁碟區

您可以在日誌事件中建立欄位的欄位索引，以進行以等式為基礎的高效率搜尋。當您接著在 CloudWatch Logs Insights 查詢中使用欄位索引時，查詢會嘗試略過處理已知不包含索引欄位的日誌事件。這可減少使用欄位索引的查詢掃描量，因此可以更快地傳回結果。這可協助您在數千個日誌群組中快速搜尋總日誌數 PB，並更快速地在相關日誌上進行搜尋。要編製索引的良好欄位是您通常需要查詢

的欄位。具有高基數值的欄位也是欄位索引的良好候選項目，因為使用這些欄位索引的查詢會更快完成，因為它會限制與目標值相符的日誌事件。

例如，假設您已建立 的欄位索引`requestId`。然後，該日誌群組 CloudWatch Logs 上包含`requestId = value`或`requestId IN [value, value, ...]`將嘗試僅處理已知包含該索引欄位和查詢值的日誌事件，且該 CloudWatch Logs 在過去已偵測到該欄位的值。

您也可以利用欄位索引來建立更多日誌群組的有效查詢。當您在查詢中使用 `filterIndex`命令而非`filter`命令時，查詢會在具有欄位索引的日誌事件上針對選取的日誌群組執行。這些查詢最多可以掃描 10，000 個您選擇的日誌群組，方法是指定最多五個日誌群組名稱字首。如果這是 CloudWatch 跨帳戶可觀測性的監控帳戶，您可以選擇所有來源帳戶或指定個別來源帳戶來選取日誌群組。

索引欄位區分大小寫。例如， 的欄位索引與包含 的日誌事件`RequestId`不相符`requestId`。

欄位索引僅支援 JSON 和服務日誌的結構化日誌格式。

CloudWatch Logs 為標準日誌類別中的所有日誌群組提供預設欄位索引。預設欄位索引會自動用於下列欄位：

- `@logStream`
- `@aws.region`
- `@aws.account`
- `@source.log`
- `@data_source_name`
- `@data_source_type`
- `@data_format`
- `traceId`
- `severityText`
- `attributes.session.id`

CloudWatch Logs 也為特定資料來源名稱和類型組合提供預設欄位索引。預設欄位索引會自動用於下列資料來源名稱和類型組合：

資料來源名稱和類型	預設欄位索引
`amazon_vpc.flow`	`action`

資料來源名稱和類型	預設欄位索引
	logStatus region flowDirection type
amazon_route53.resolver_query	query_type transport rcode
aws_waf.access	action httpRequest.country
aws_cloudtrail.data aws_cloudtrail.management	eventSource eventName awsRegion userAgent errorCode eventType managementEvent readOnly eventCategory requestId

預設欄位索引是附加於您在政策中定義的任何自訂欄位索引。預設欄位索引不會計入您的欄位索引配額。

CloudWatch Logs 只會將建立索引政策後擷取的日誌事件編製索引。它不會為建立政策之前擷取的日誌事件編製索引。建立欄位索引之後，每個相符的日誌事件都會從日誌事件的擷取時間開始保持索引 30 天。

> **Note**
>
> 如果您在監控帳戶中建立欄位索引政策，該政策不會用於連結來源帳戶中的日誌群組。欄位索引政策僅適用於建立該政策的帳戶。

本節中的其餘主題說明如何建立欄位索引。如需有關在查詢中參考欄位索引的資訊，請參閱 filterIndex和 篩選條件。

主題

- 欄位索引語法和配額
- 建立帳戶層級欄位索引政策
- 建立日誌群組層級欄位索引政策
- 建立查詢時的日誌群組選取選項
- 刪除欄位索引政策的效果

欄位索引語法和配額

您可以透過建立欄位索引政策來建立欄位索引。您可以建立套用至整個帳戶的帳戶層級索引政策，也可以建立僅適用於單一日誌群組的政策。對於整個帳戶的索引政策，您可以有一個適用於帳戶中所有日誌群組的索引政策。您也可以建立套用到帳戶中日誌群組子集的帳戶層級索引政策，由其日誌群組名稱的字首選取。如果您在同一個帳戶中有多個帳戶層級政策，則這些政策的日誌群組名稱字首無法重疊。同樣地，您可以建立套用至特定資料來源名稱和類型組合的帳戶層級索引政策。每個資料來源名稱和類型組合只能建立一個帳戶政策。

日誌群組層級欄位索引政策會覆寫帳戶層級欄位索引政策：整體套用至日誌群組 （例如，沒有選取條件或日誌群組名稱字首型選取條件的帳戶層級政策）。除了符合整體日誌群組的政策之外，還會套用符合日誌事件層級的帳戶層級政策 （例如，指定資料來源名稱和類型組合）。如果您建立日誌群組層級索引政策，則該日誌群組不會使用與日誌群組層級相符的帳戶層級政策。

日誌事件與欄位索引名稱的比對會區分大小寫。例如， 的欄位索引與包含 的日誌事件`RequestId`不相符`requestId`。

您最多可以有 40 個帳戶層級索引政策，其中 20 個政策可以使用日誌群組名稱字首選擇條件，而 20 個政策可以使用資料來源型選擇條件。如果您有多個帳戶層級索引政策篩選為日誌群組名稱字首，則其中沒有任何兩個政策可以使用相同或重疊的日誌群組名稱字首。例如，如果您有一個政策篩選為以 開頭的日誌群組`my-log`，則您無法將另一個欄位索引政策篩選為 `my-logpprod`或 `my-logging`。同樣地，如果您有多個帳戶層級索引政策篩選為資料來源名稱和類型組合，則其中沒有任何兩個政策可以使用相同的資料來源名稱和類型。例如，如果您有一個政策篩選為資料來源名稱`amazon_vpc`和資料來源類型`flow`，則無法使用此組合建立另一個政策。

如果您的帳戶層級索引政策沒有名稱字首且適用於所有日誌群組，則無法建立具有日誌群組名稱字首篩選條件的其他帳戶層級索引政策；您可以建立使用資料來源名稱和類型篩選條件的帳戶層級索引政策。

每個索引政策都有下列配額和限制：

- 政策最多可包含 20 個欄位。
- 每個欄位名稱最多可包含 100 個字元。
- 若要在以 開頭的日誌群組中建立自訂欄位的索引`@`，您必須在欄位名稱的`@`開頭指定具有額外 的欄位。例如，如果您的日誌事件包含名為 的欄位`@userId`，您必須指定 `@@userId` 為此欄位建立索引。

對於具有資料來源名稱和類型型選取條件的帳戶層級索引政策，會套用額外的限制：所有欄位都必須是基本資料類型，結構僅支援巢狀基本。

產生的欄位和預留欄位

CloudWatch Logs Insights 會在每個日誌事件中自動產生系統欄位。這些產生的欄位字首為 `@`，如需所產生欄位的詳細資訊，請參閱 支援的日誌和探索的欄位。

在這些產生的欄位中，支援使用下列項目做為欄位索引：

- `@logStream`
- `@ingestionTime`
- `@requestId`
- `@type`
- `@initDuration`
- `@duration`
- `@billedDuration`

- @memorySize
- @maxMemoryUsed
- @xrayTraceId
- @xraySegmentId

若要為這些產生的欄位編製索引，您不需要在指定@時新增額外的 ，因為您必須對開頭為 的自訂欄位執行此操作@。例如，若要建立 的欄位索引@logStream，只要指定 @logStream做為欄位索引即可。

CloudWatch Logs 為標準日誌類別中的所有日誌群組提供預設欄位索引。預設欄位索引會自動用於下列欄位：

- @logStream
- @aws.region
- @aws.account
- @source.log
- @data_source_name
- @data_source_type
- @data_format
- traceId
- severityText
- attributes.session.id

CloudWatch Logs 也為特定資料來源名稱和類型組合提供預設欄位索引。預設欄位索引會自動用於下列資料來源名稱和類型組合：

資料來源名稱和類型	預設欄位索引
amazon_vpc.flow	action logStatus region flowDirection

資料來源名稱和類型	預設欄位索引
	type
amazon_route53.resolver_query	query_type transport rcode
aws_waf.access	action httpRequest.country
aws_cloudtrail.data aws_cloudtrail.management	eventSource eventName awsRegion userAgent errorCode eventType managementEvent readOnly eventCategory requestId

預設欄位索引是附加於您在政策中定義的任何自訂欄位索引。預設欄位索引不會計入您的欄位索引配額。

JSON 日誌中的子欄位和陣列欄位

您可以在 JSON 日誌中為巢狀子欄位或陣列欄位的欄位編製索引。

例如，您可以在此日誌的 欄位內建立accessKeyId子userIdentity欄位的索引：

```
{
    "eventVersion": "1.0",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE_PRINCIPAL_ID",
        "arn": "arn: aws: iam: : 123456789012: user/Alice",
        "accessKeyId": "11112222",
        "accountId": "123456789012",
        "userName": "Alice"
    },
    "eventTime": "2014-03-06T21: 22: 54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "ec2-api-tools1.6.12.2",
    "requestParameters": {
        "instancesSet": {
            "items": [{
                "instanceId": "i-abcde123",
                "currentState": {
                    "code": 0,
                    "name": "pending"
                },
                "previousState": {
                    "code": 80,
                    "name": "stopped"
                }
            }]
        }
    }
}
```

若要建立此欄位，您可以在建立欄位索引和在查詢中指定欄位索引時，使用點符號 (`userIdentity.accessKeyId`) 來參考它。查詢可能如下所示：

```
fields @timestamp, @message
| filterIndex userIdentity.accessKeyId = "11112222"
```

在先前的範例事件中， `instanceId` 欄位位於 陣列中 `requestParameters.instancesSet.items` 若要在建立欄位索引時和查詢時都代表此欄位，請將

其稱為 requestParameters.instancesSet.items.0.instanceId 0，表示該欄位在陣列中的位置。

然後，此欄位的查詢可以是下列項目：

```
fields @timestamp, @message
| filterIndex requestParameters.instancesSet.items.0.instanceId="i-abcde123"
```

建立帳戶層級欄位索引政策

使用本節中的步驟來建立欄位索引政策，該政策會套用至帳戶中的所有日誌群組，或套用至具有以相同字串開頭之日誌群組名稱的多個日誌群組。

建立帳戶層級欄位索引政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇設定，然後選擇日誌索引標籤。
3. 在帳戶層級索引政策區段中，選擇管理。
4. 選擇建立索引政策。
5. 在政策名稱中，輸入新政策的名稱。
6. 對於選取政策範圍，請執行下列其中一項操作：
   - 選擇所有標準日誌群組，讓索引政策套用至帳戶中的所有標準類別日誌群組。
   - 依字首比對選擇日誌群組，將政策套用至名稱以相同字串開頭的日誌群組子集。然後，在輸入字首名稱中輸入這些日誌群組的字首。

     輸入字首後，您可以選擇預覽字首相符的日誌群組，以確認字首符合您預期的日誌群組。

     選擇依資料來源記錄資料，將政策套用至特定資料來源名稱和類型組合。然後，您可以從下拉式選單中選取資料來源和資料類型。

     選取資料來源名稱和類型後，您可以選取取得欄位，將可用欄位、包含的日誌群組，以及預設和自訂欄位索引等相關資訊填入設定欄位索引和面向區段。
7. 針對自訂索引欄位組態，選擇新增欄位路徑以輸入要索引的第一個欄位。

   然後輸入要用作欄位名稱值的字串，或從下拉式功能表中選取欄位。這必須與日誌事件中顯示的內容完全相符。例如，如果您的日誌事件包含 requestId，則必須requestId在此處輸入 。RequestId、 requestID和 request Id不相符。

如果您想要為以 @ 字元開頭的自訂日誌欄位編製索引，您必須在輸入索引字串時包含額外的@字元。例如，如果您有自訂日誌欄位 @emailname，@@emailname請在新增欄位路徑方塊中輸入 。

您也可以為 CloudWatch Logs 自動產生的 @ingestionTime和 @logStream 欄位建立索引。如果您這麼做，則不需要在指定@時新增額外的 。

8. （選用） 除了指定欄位路徑之外，您還可以選取設定為面向，以將欄位建立為面向。
9. 重複上一個步驟，新增多達 20 個欄位索引。
10. 當您完成時，請選擇 Create (建立)。

建立日誌群組層級欄位索引政策

使用本節中的步驟來建立套用至單一日誌群組的欄位索引政策。

建立日誌群組層級欄位索引政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中依序選擇 Logs (日誌) > Log groups (日誌群組)。
3. 選擇日誌群組的名稱。
4. 選擇欄位索引索引標籤。
5. 選擇管理此日誌群組的欄位索引
6. 針對管理日誌群組層級欄位索引，選擇新增欄位路徑以輸入要編製索引的第一個欄位。

   然後輸入要用作欄位名稱值的字串。這必須與日誌事件中顯示的內容完全相符。例如，如果您的日誌事件包含 requestId，則必須requestId在此處輸入 。RequestId、 requestID和request Id不相符。

   如果您想要為以 @ 字元開頭的自訂日誌欄位編製索引，您必須在輸入索引字串時包含額外的@字元。例如，如果您有自訂日誌欄位 @emailname，@@emailname請在新增欄位路徑方塊中輸入 。

   您也可以為 CloudWatch Logs 自動產生的 @ingestionTime和 @logStream 欄位建立索引。如果您這麼做，則不需要在指定@時新增額外的 。
7. （選用） 除了指定欄位路徑之外，您還可以選取設定為面向，將欄位建立為面向。
8. 重複上一個步驟，新增多達 20 個欄位索引。

9. 完成後，請選擇 Save (儲存)。

建立查詢時的日誌群組選取選項

本節說明您可以選取要包含在查詢中的日誌群組的各種方式。

在主控台中選取查詢的日誌群組

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌，然後選擇 Logs Insights。
3. 選取您要用於此查詢的查詢語言。您可以選擇：Logs Insights QL、OpenSearch PPL 或 OpenSearch SQL。
4. 有三種方式可以選取查詢的日誌群組：
   - 使用日誌群組名稱方塊。這是預設選擇方法。您可以使用此方法輸入最多 50 個日誌群組名稱。如果這是 CloudWatch 跨帳戶觀察功能中的監控帳戶，您可以在來源帳戶和監控帳戶中選取日誌群組。單一查詢可以一次查詢來自不同帳戶的日誌。
   - 使用日誌群組條件區段。在本節中，您可以根據日誌群組名稱的字首選擇日誌群組。您可以在一個查詢中包含最多五個字首。將會選取名稱中具有這些字首的日誌群組。或者，所有日誌群組選項會從帳戶選取所有日誌群組。
   - 如果這是 CloudWatch 跨帳戶可觀測性的監控帳戶，您可以在帳戶下拉式功能表中選取所有帳戶，從所有連結帳戶中選取日誌群組。或者，您可以個別選取此查詢應包含哪些帳戶。

   如果您的選擇符合超過 10，000 個日誌群組，您會看到錯誤，提示您縮小選擇範圍。
5. 查詢的預設日誌類別為標準。您可以使用日誌類別將其變更為不常存取。

使用 AWS CLI

若要在從命令列啟動查詢時進行這些類型的選擇，您可以在查詢中使用 `source`命令。如需詳細資訊和範例，請參閱 SOURCE。

刪除欄位索引政策的效果

如果您刪除已生效一段時間的欄位索引政策，會發生下列情況：

- 在刪除政策後最多 30 天內，查詢仍然可以從索引日誌事件中受益。
- 如果您刪除日誌群組層級索引政策，且已有適用於該日誌群組的帳戶層級政策，則帳戶層級政策最終會套用至該日誌群組。

使用面向來分組和探索日誌

面向有助於分析日誌，因為它們可讓您以互動方式篩選和分組資料，而無需執行查詢。面向是日誌中的欄位（例如 ServiceName或 StatusCode)，可啟用跨日誌群組的篩選、彙總和分析。您可以在 CloudWatch Logs Insights 主控台中檢視面向欄位的清單，以及根據您選取的時間範圍，每個面向值的日誌事件計數。當您選取不同的面向和值時，面向值和計數會即時更新，讓您以互動方式探索日誌。

每個面向都會根據選取的時間範圍和查詢範圍，顯示日誌中欄位自動擷取的可用值和計數，並保留 30 天。顯示的面向計數為近似值。您可以使用資料來源名稱或資料來源類型等預設面向來探索日誌，或在日誌中的任何欄位上建立自訂面向。資料來源名稱是產生日誌的 AWS 服務或應用程式（例如 Route 53、Amazon VPC 或 CloudTrail)，而資料來源類型是該服務產生的特定日誌類型。預設面向由 CloudWatch 建立，並包含 @aws.region、@data_source_type、 @data_source_name和 @data_format。如需詳細資訊，請參閱日誌管理。面向僅適用於帳戶中導入的日誌。如果您已設定跨帳戶可觀測性，監控帳戶將無法根據來源帳戶的日誌檢視面向。

若要建立其他面向，請選取日誌中與您的故障診斷相關的欄位，並使用索引政策進行設定。對於自訂面向，我們建議在低基數欄位上建立它們（每天小於 100 個唯一值的欄位，例如狀態和 ApplicationName)。每天有超過 100 個唯一值的構面會分類為高基數，且不會顯示這些構面的值。選取一或多個面向，然後按一下以跨日誌執行查詢。

若要開始使用 CloudWatch Logs Insights 中的面向：

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌，然後選擇 Logs Insights。
3. （選用）使用時間範圍選擇器來選取您要分析的時段。對於選取的時間範圍，可用的面向和值會顯示在面板中。
4. 選取構面以探索您的資料，並查看各個構面中值分佈的即時更新。

   不會顯示具有超過 100 個唯一值的面向。若要查詢特定值，請改為在查詢中使用篩選條件。

執行面向型查詢

1. 跨面向選取一或多個值。
2. 事件計數會根據選取的面向和值更新。
3. 選取面向值時，查詢範圍會更新以反映選擇。
4. 選取面向值後，按一下執行以執行查詢。

5. 每個面向支援的唯一值數目上限為 100。例如，如果面向有 100 個以上的值，則所有計數會顯示為 "-"，表示值未知。

儲存面向型查詢

1. 使用一或多個面向值建立查詢。
2. 其餘步驟與儲存 Logs Insights 查詢相同。請參閱儲存 CloudWatch Logs Insights 查詢。
3. 您儲存的查詢可在儲存的查詢區段中取得。當您擷取已儲存的查詢時，它會自動包含用於查詢的面向和值，讓您輕鬆分析日誌。

建立帳戶層級面向

1. 若要建立面向，您必須先建立 欄位做為索引，並將其設定為面向。在導覽窗格中，選擇設定、日誌、帳戶層級索引政策。或者，您可以在面向面板上選取管理面向。
2. 選擇建立新的索引政策。如需建立索引政策的詳細資訊，請參閱 建立帳戶層級欄位索引政策。
3. 若要建立面向，請在索引政策建立頁面中勾選將選定欄位設為面向。

使用 APIs面向管理

您可以使用欄位索引政策來完成面向管理。如需詳細資訊，請參閱 `field index` API。 APIs

欄位索引 APIs

否。	名稱	描述
1	PutIndexPolicy	建立或更新特定日誌群組的欄位索引政策
2	PutAccountPolicy	建立帳戶層級資料保護政策、訂閱篩選條件政策、欄位索引政策、轉換器政策或指標擷取政策，適用於帳戶中的所有日誌群組或日誌群組子集
3	DeleteIndexPolicy	刪除套用至單一日誌群組的日誌群組層級欄位索引政策
4	DeleteAccountPolicy	刪除 CloudWatch Logs 帳戶政策

模式分析

CloudWatch Logs Insights 會使用機器學習演算法來尋找模式。模式是在您的日誌欄位之間重複出現的共用文字結構。當您檢視查詢的結果時，您可以選擇模式索引標籤，以查看 CloudWatch Logs 根據結果範例找到的模式。或者，您可以將 `pattern`命令附加至查詢，以分析整組相符日誌事件中的模式。

模式有助於分析大型日誌集，因為大量日誌事件通常可以壓縮為幾個模式。

請考慮以下三個日誌事件的範例。

```
2023-01-01 19:00:01 [INFO] Calling DynamoDB to store for resource id 12342342k124-12345
2023-01-01 19:00:02 [INFO] Calling DynamoDB to store for resource id 324892398123-12345
2023-01-01 19:00:03 [INFO] Calling DynamoDB to store for resource id 3ff231242342-12345
```

在先前的範例中，這三個日誌事件都遵循一個模式：

```
<Time-1> [INFO] Calling DynamoDB to store for resource id <ID-2>
```

模式內的欄位稱為字符。模式內不同的欄位，例如請求 ID 或時間戳記，都是動態字符。每個動態字符都由 表示<*string-number*>。*##*是字符所代表資料類型的描述。此*##*顯示與其他動態字符相比，此字符在模式中的顯示位置。

動態字符的常見範例包括錯誤代碼、時間戳記和請求 IDs。權杖值代表動態權杖的特定值。例如，如果動態字符代表 HTTP 錯誤代碼，則字符值可以是 `501`。

模式偵測也用於 CloudWatch Logs 異常偵測器並比較功能。如需詳細資訊，請參閱日誌異常偵測及比較（差異）與先前的時間範圍。

模式分析入門

模式偵測會在任何 CloudWatch Logs Insights 查詢中自動執行。不包含 `pattern`命令的查詢會在結果中同時取得日誌事件和模式。

如果您在查詢中包含 `pattern`命令，則會對整組相符的日誌事件執行模式分析。這可提供更準確的模式結果，但當您使用 `pattern`命令時，不會傳回原始日誌事件。當查詢不包含 時`pattern`，模式結果會根據前 1000 個傳回的日誌事件，或是您在查詢中使用的限制值。如果您在查詢`pattern`中包含 ，則模式索引標籤中顯示的結果會衍生自查詢相符的所有日誌事件。

在 CloudWatch Logs Insights 中開始使用模式分析

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。

2. 在導覽窗格中，選擇日誌，然後選擇 Logs Insights。

   在 Logs Insights (日誌洞察) 頁面上，查詢編輯器包含會傳回 20 筆最新日誌事件的預設查詢。

3. 移除查詢方塊中的`| limit 20`行，讓查詢如下所示：

```
fields @timestamp, @message, @logStream, @log
| sort @timestamp desc
```

4. 在選取日誌群組 (Select log group) 下拉式清單中，選擇要查詢的一或多個日誌群組。
5. (選用) 使用時間間隔選擇器，選取您要查詢的時間段。

   您可以選擇 5 分鐘和 30 分鐘間隔；1 小時、3 小時和 12 小時間隔；或自訂時間範圍。

6. 選擇執行查詢以開始查詢。

   當查詢完成執行時，日誌索引標籤會顯示查詢傳回的日誌事件資料表。表格上方是有關有多少筆記錄符合查詢的訊息，類似於顯示 71，101 筆記錄中有 10，000 筆符合。

7. 選擇模式索引標籤。
8. 資料表現在會顯示查詢中找到的模式。由於查詢不包含`pattern`命令，此標籤只會顯示 10，000 個日誌事件中探索到的模式，這些事件會顯示在日誌標籤的表格中。

   對於每個模式，會顯示下列資訊：

   - 模式，每個動態字符都顯示為 <*string-number*>。*##*是字符所代表資料類型的描述。此*##*顯示與其他動態字符相比，此字符在模式中的顯示位置。
   - 事件計數，這是模式出現在查詢日誌事件中的次數。選擇事件計數欄標題，依頻率排序模式。
   - 事件比率，即包含此模式的查詢日誌事件百分比。
   - 嚴重性類型，將是下列其中一項：
     - 如果模式包含錯誤一詞，則為錯誤。
     - 如果模式包含警告一詞，但不包含錯誤，請警告 。
     - 如果模式不包含警告或錯誤，則為 INFO。

   選擇嚴重性資訊欄標題，依嚴重性排序模式。

9. 現在變更查詢。將查詢中的`| sort @timestamp desc`行取代為`| pattern @message`，讓完整的查詢如下所示：

```
fields @timestamp, @message, @logStream, @log
| pattern @message
```

10. 選擇 Run query (執行查詢)。

    查詢完成時，日誌索引標籤中沒有結果。不過，根據查詢的日誌事件總數，模式索引標籤可能列出更多模式。

11. 無論您是否包含在查詢pattern中，都可以進一步檢查查詢傳回的模式。若要這樣做，請在檢查欄中選擇其中一個模式的圖示。

    模式檢查窗格隨即出現，並顯示下列項目：

    - 模式。在模式中選取權杖，以分析該權杖的值。
    - 顯示查詢時間範圍內模式出現次數的長條圖。這可協助您識別有趣的趨勢，例如模式的出現突然增加。
    - 日誌範例索引標籤會顯示一些符合所選模式的日誌事件。
    - 如果您已選取動態權杖，權杖值索引標籤會顯示所選動態權杖的值。

    > **Note**
    >
    > 每個字符最多擷取 10 個字符值。字符計數可能不精確。CloudWatch Logs 使用機率計數器來產生字符計數，而不是絕對值。

    - 相關模式索引標籤會顯示與您檢查的模式幾乎同時經常發生的其他模式。例如，如果ERROR訊息的模式通常伴隨另一個標記為 的日誌事件INFO和其他詳細資訊，則此模式會顯示在此處。

模式命令的詳細資訊

本節包含有關 pattern命令及其用途的更多詳細資訊。

- 在先前的教學課程中，我們在新增 時移除 sort命令，pattern因為如果查詢在sort命令之後包含pattern命令，則查詢無效。在 pattern之前擁有 是有效的sort。

  如需pattern語法的詳細資訊，請參閱 pattern。
- 當您pattern在查詢中使用 時， @message 必須是在 pattern命令中選取的其中一個欄位。
- 您可以在filter命令之前包含 pattern命令，只讓篩選過的日誌事件集做為模式分析的輸入。
- 若要查看特定欄位的模式結果，例如衍生自parse命令的欄位，請使用 pattern @fieldname。
- 使用非日誌輸出的查詢，例如使用 stats命令的查詢，不會傳回模式結果。

儲存並重新執行 CloudWatch Logs Insights 查詢

建立查詢之後，可以儲存該查詢，以便之後再次執行。查詢會儲存在資料夾結構中，因此您可以組織它們。每個區域和每個帳戶最多可儲存 1000 個查詢。

查詢會儲存在區域特定層級，而非使用者特定層級。如果您建立並儲存查詢，在同一區域中具有 CloudWatch Logs 存取權的其他使用者可以查看該區域中所有已儲存的查詢及其資料夾結構。

若要儲存查詢，您必須登入具有許可 `logs:PutQueryDefinition` 的角色。若要查看已儲存查詢的清單，您必須登入具有許可 `logs:DescribeQueryDefinitions` 的角色。

> **Note**
>
> 您可以使用 參數建立和儲存查詢 — 具有具名預留位置的可重複使用範本。與其使用不同的值儲存相同查詢的多個變化，請建立一個範本，並在執行時提供不同的參數值。此功能目前僅支援使用 Logs Insights 查詢語言的查詢。如需詳細資訊，請參閱搭配參數使用已儲存的查詢。

Console

儲存查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 在查詢編輯器中，建立查詢。
4. 選擇儲存。
5. 輸入查詢的名稱。
6. (選用) 選擇您要儲存查詢的資料夾。選取 Create new (新建) 以建立資料夾。如果您建立新資料夾，您可以在資料夾名稱中使用斜線 (/) 字元，以定義資料夾結構。例如，命名新資料夾 **`folder-level-1/folder-level-2`** 會建立名為 **`folder-level-1`** 的頂層資料夾，該資料夾中會有另一個資料夾名為 **`folder-level-2`**。查詢會儲存在 **`folder-level-2`** 中。
7. (選用) 變更查詢的日誌群組或查詢文字。
8. （選用）若要在查詢中使用參數，請遵循下列其他步驟：
   a. 將參數新增至查詢。使用`{{parameter}}`語法（參數名稱前後的雙括號）將靜態值取代為預留位置。

      範例：具有靜態值的原始查詢：

```
fields @timestamp, @message
| filter level = "Error"
| filter applicationName = "OrderService"
```

使用參數更新查詢：

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

b. 定義查詢中使用的參數。針對每個預留位置參數，指定：

- 名稱：必須與預留位置名稱完全相符（例如，logLevel、applicationName)。
- 預設值（選用）：如果未提供參數值，要使用的值。
- 描述（選用）：說明 參數的用途。

c. 您可以使用具有 $字首的查詢名稱，並將參數名稱做為索引鍵/值對傳遞，來執行具有參數的查詢。如需詳細資訊，請參閱執行已儲存的查詢。

9. 選擇儲存。

AWS CLI

若要儲存查詢，請使用 put-query-definition：

```
aws logs put-query-definition \
  --name "ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = \"ERROR\"" \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

（選用）若要使用參數儲存查詢，請新增 --parameters選項，並在查詢字串中使用{{parameterName}}預留位置：

```
aws logs put-query-definition \
  --name "ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = {{logLevel}} | filter
 applicationName = {{applicationName}}" \
  --parameters '[{"name":"logLevel","defaultValue":"ERROR","description":"Log level
 to filter"},
```

```
{"name":"applicationName","defaultValue":"OrderService","description":"Application
 name to filter"}]' \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

若要將查詢儲存在資料夾中，請在查詢名稱前面加上資料夾路徑：

```
aws logs put-query-definition \
  --name "my-folder/ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = {{logLevel}}" \
  --parameters '[{"name":"logLevel","defaultValue":"ERROR","description":"Log level
 to filter"}]' \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

API

若要儲存查詢，請呼叫 PutQueryDefinition：

```
{
  "name": "ErrorsByLevel",
  "queryString": "fields @timestamp, @message | filter level = \"ERROR\"",
  "logGroupNames": ["/aws/lambda/my-function"]
}
```

（選用）若要使用參數儲存查詢，請在查詢字串中包含 parameters 欄位並使用{{parameterName}}預留位置：

```
{
  "name": "ErrorsByLevel",
  "queryString": "fields @timestamp, @message | filter level = {{logLevel}} | filter
 applicationName = {{applicationName}}",
  "logGroupNames": ["/aws/lambda/my-function"],
  "parameters": [
    {
      "name": "logLevel",
      "defaultValue": "ERROR",
      "description": "Log level to filter"
    },
    {
      "name": "applicationName",
      "defaultValue": "OrderService",
```

```
      "description": "Application name to filter"
    }
  ]
}
```

> **Tip**
>
> 您可以使用 PutQueryDefinition 建立已儲存查詢的資料夾。若要為儲存的查詢建立資料夾，請使用正斜線 (/) ，在所需查詢名稱前加上想要的資料夾名稱：<*folder-name*>/<*query-name*>。如需有關此動作的詳細資訊，請參閱 PutQueryDefinition。

Console

執行已儲存的查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 選擇右側的 Queries (查詢)。
4. 從已儲存的查詢清單中選取您的查詢。查詢文字會出現在查詢編輯器中。
5. （選用）若要搭配參數使用查詢：
   a. 在已儲存查詢側邊面板中選擇查詢名稱旁的 + 圖示。
   b. 具有參數的查詢會出現在查詢編輯器中。例如，如果您選擇 旁的 + 圖示ErrorsByLevel，查詢編輯器會填入： `$ErrorsByLevel(level=, applicationName=)`
   c. 提供參數 (level、applicationName) 的值並執行查詢。例如：`$ErrorsByLevel(level= "ERROR", applicationName= "OrderService")`
6. 選擇執行。

AWS CLI

使用參數執行已儲存的查詢

`start-query` 搭配 `$QueryName()` 語法使用：

```
aws logs start-query \
```

```
  --log-group-names "/aws/lambda/my-function" \
  --start-time 1707566400 --end-time 1707570000 \
  --query-string '$ErrorsByLevel(level= "ERROR", applicationName= "OrderService")' \
  --region us-east-1
```

API

使用參數執行已儲存的查詢

使用 `queryString` 欄位中的`$QueryName()`語法呼叫 StartQuery：

```
{
  "logGroupNames": ["/aws/lambda/my-function"],
  "startTime": 1707566400,
  "endTime": 1707570000,
  "queryString": "$ErrorsByLevel(level=\"ERROR\", applicationName= \"OrderService
\")"
}
```

儲存新版本的已儲存查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 選擇右側的 Queries (查詢)。
4. 從 Saved queries (已儲存的查詢) 清單中選取查詢。它會出現在查詢編輯器中。
5. 修改查詢。如果您需要執行該功能以檢查您的工作，請選擇 Run query (執行查詢)。
6. 當您準備好儲存新版本，請選擇 Actions (動作)、Save as (另存新檔)。
7. 輸入查詢的名稱。
8. (選用) 選擇您要儲存查詢的資料夾。選取 Create new (新建) 以建立資料夾。如果您建立新資料夾，您可以在資料夾名稱中使用斜線 (/) 字元，以定義資料夾結構。例如，命名新資料夾 **`folder-level-1/folder-level-2`** 會建立名為 **`folder-level-1`** 的頂層資料夾，該資料夾中會有另一個資料夾名為 **`folder-level-2`**。查詢會儲存在 **`folder-level-2`** 中。
9. (選用) 變更查詢的日誌群組或查詢文字。
10. 選擇儲存。

若要刪除查詢，您必須登入具備 `logs:DeleteQueryDefinition` 許可的角色。

編輯或刪除已儲存的查詢

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 選擇右側的 Queries (查詢)。
4. 從 Saved queries (已儲存的查詢) 清單中選取查詢。它會出現在查詢編輯器中。
5. 選擇 Actions (動作)、Edit (編輯) 或 Actions (動作)、Delete (刪除)。

搭配參數使用已儲存的查詢

具有參數的已儲存查詢是具有具名預留位置的可重複使用查詢範本。您可以儲存範本並在執行查詢時提供不同的參數值，而不是維護幾乎相同的查詢的多個副本。只有 CloudWatch Logs Insights 查詢語言才支援參數。

運作方式

儲存查詢時，預留位置會識別您可以在查詢執行時間提供的值。預留位置使用 `{{parameterName}}` 語法。以下是名為 且`ErrorsByLevel`具有兩個參數 `logLevel`和 的已儲存查詢範例`applicationName`。

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

若要執行儲存的查詢，您可以使用字首為 的查詢名稱來叫用它，`$`並傳遞參數值。CloudWatch Logs Insights 查詢引擎會取代每個預留位置。如果參數包含預設值，則如果未提供其他值，則會使用這些值。

```
# Run query by using query name and passing parameter values explicitly
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService")

# Run query without specifying parameter values - default values are used in this case.
$ErrorsByLevel()
```

包含空格或特殊字元的已儲存查詢名稱需要用反引號括住：

```
$`Errors By Level`(logLevel = "WARN")
```

具有參數的已儲存查詢範例

新增結果限制做為參數

查詢名稱：ErrorsByLevel含參數 logLevel（預設值："ERROR")、 applicationName （預設值："OrderService") 和 maxResults（預設值：50)

```
fields @timestamp, @message, @logStream
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
| sort @timestamp desc
| limit {{maxResults}}
```

```
# Run the query using the query name and passing parameter values
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService", maxResults = 100)
```

搭配參數使用多個已儲存的查詢

以下範例使用 ErrorsByLevel和第二個儲存的查詢RecentN，其定義為 sort @timestamp desc | limit {{count}}（使用參數 count，預設 20)。CloudWatch Logs Insights 查詢引擎會在執行前展開每個查詢。

```
# Using multiple queries with parameters in sequence
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService")
| $RecentN(count = 10)

# Each of the queries is expanded, resulting in the following query when it is run.
fields @timestamp, @message
| filter level = "WARN"
| filter applicationName = "OrderService"
| sort @timestamp desc
| limit 10
```

配額和錯誤處理

> **Note**
>
> 每個儲存的查詢最多可有 20 個參數。

展開的查詢字串不能超過 10，000 個字元。參數名稱必須以字母或底線開頭。儲存的查詢無法參考另一個儲存的查詢（不支援巢狀調用）。

常見錯誤

錯誤	原因
只有 CWLI 查詢語言才支援參數	只有 CloudWatch Logs Insights 查詢語言才支援參數。
queryString 中找不到必要的參數	中的參數名稱在查詢字串{{placeholder}} 中--parameters 沒有相符項目。
參數計數超過上限 20	儲存的查詢目前僅支援 20 個參數。
重複的參數名稱	查詢定義在 中有重複的參數parameters 。

> **Note**
>
> 若要使用參數建立或更新已儲存的查詢，您需要 logs:PutQueryDefinition許可。若要執行一個，您需要 logs:StartQuery和 logs:DescribeQueryDefinitions。

將查詢新增到儀表板或匯出查詢結果

執行查詢之後，您可以將查詢新增至 CloudWatch 儀表板，或將結果複製到剪貼簿。

每次載入儀表板和每次儀表板重新整理時，會執行新增到儀表板的查詢。這些查詢會計入 100 個並行 CloudWatch Logs Insights 查詢的限制。

將查詢結果新增到儀表板

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 選擇一或多個日誌群組並執行查詢。
4. 選擇 Add to dashboard (新增至儀表板)。
5. 選取儀表板，或選擇 Create new (新建)，為查詢結果建立儀表板。
6. 選取用於查詢結果的 Widget 類型。

7. 輸入 Widget 的名稱。
8. 選擇 Add to dashboard (新增至儀表板)。

將查詢結果複製到剪貼簿或下載查詢結果

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 選擇一或多個日誌群組並執行查詢。
4. 選擇 Export results (匯出結果)，然後選擇您需要的選項。

檢視執行中的查詢或查詢歷史記錄

您可以檢視目前進行中的查詢，以及您的最新查詢歷史記錄。

目前執行中的查詢包括您已新增到儀表板的查詢。每個帳戶僅限 100 個並行 CloudWatch Logs Insights 查詢，包括新增至儀表板的查詢。此外，您可以為 OpenSearch Service PPL 或 OpenSearch Service SQL 執行 15 個並行查詢。

檢視您的最新查詢歷史記錄

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
3. 如果您使用新設計的 CloudWatch Logs 主控台，請選擇 History (歷史記錄)。如果您是使用舊設計，請選擇 Actions (動作)、View query history for this account (檢視此帳戶的查詢歷程記錄)。

   隨即顯示您的最新查詢清單。您可以選取查詢，並選擇 Run (執行)，以再次執行任一查詢。

   在 Status (狀態) 下，CloudWatch Logs 會將目前執行中的任何查詢顯示為 In progress (進行中)。

使用 加密查詢結果 AWS Key Management Service

根據預設，CloudWatch Logs 會使用 CloudWatch Logs 伺服器端預設的加密方法，加密您 CloudWatch Logs Insights 查詢所儲存的結果。您可以選擇改用 AWS KMS 金鑰來加密這些結果。如果您將 AWS KMS 金鑰與加密結果建立關聯，則 CloudWatch Logs 會使用該金鑰來加密帳戶中所有查詢的儲存結果。

如果您之後取消金鑰與查詢結果的關聯，CloudWatch Logs 會在日後的查詢中重新使用預設的加密方法。不過，在金鑰仍有關聯時所執行的查詢，依然會使用該金鑰來加密。由於 CloudWatch Logs 仍可繼續參照該金鑰，因此依然可在 KMS 金鑰解除關聯後傳回那些結果。然而，若之後將金鑰停用，CloudWatch Logs 就無法讀取使用該金鑰加密的日誌。

> **Important**
>
> CloudWatch Logs 僅支援對稱 KMS 金鑰。切勿使用非對稱金鑰加密查詢結果。如需詳細資訊，請參閱使用對稱和非對稱金鑰。

限制

- 若要執行下列步驟，您必須擁有下列許可：`kms:CreateKey`、`kms:GetKeyPolicy` 和 `kms:PutKeyPolicy`。
- 建立或取消金鑰與查詢結果的關聯後，操作會在 5 分鐘內生效。
- 如果您撤銷 CloudWatch Logs 對已關聯金鑰的存取權，或刪除所關聯的 KMS 金鑰，您將無法再擷取 CloudWatch Logs 中的加密資料。
- 您無法使用 CloudWatch 主控台為金鑰建立關聯，必須使用 AWS CLI 或 CloudWatch Logs API 才行。

步驟 1：建立 AWS KMS key

若要建立 KMS 金鑰，請使用以下 create-key 命令：

```
aws kms create-key
```

輸出包含金鑰 ID 和金鑰的 Amazon Resource Name (ARN)。下列為範例輸出：

```
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
```

```
        "KeyState": "Enabled",
        "CreationDate": 1478910250.94,
        "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-
e40cb0d29f59",
        "AWSAccountId": "123456789012",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

步驟 2：設定 KMS 金鑰許可

根據預設，所有 KMS 金鑰皆屬私有。只有資源擁有者可以使用它來加密和解密資料。然而，資源擁有者可以授予其他使用者和資源存取金鑰的許可。在此步驟中，您會給予 CloudWatch Logs 服務主體使用金鑰的許可。此服務主體必須位於存放金鑰的相同 AWS 區域中。

根據最佳實務，建議您將金鑰的使用限制為您指定的 AWS 帳戶。

首先，使用以下 get-key-policy 命令，將 KMS 金鑰的預設政策儲存為 `policy.json`：

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

在文字編輯器中開啟 `policy.json` 檔案，並從下列其中一個陳述式中加入區段 (以粗體顯示)。使用逗號從新陳述式中分隔現有陳述式。這些陳述式使用`Condition`區段來增強 AWS KMS 金鑰的安全性。如需詳細資訊，請參閱AWS KMS 金鑰和加密內容。

此範例中的 `Condition`區段會將 AWS KMS 金鑰的使用限制在指定帳戶中的 CloudWatch Logs Insights 查詢結果。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.region.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt*",
                "kms:Decrypt*",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:Describe*"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                "aws:SourceArn": "arn:aws:logs:us-east-1:111122223333:query-
result:*"
                },
                "StringEquals": {
                "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

最後，使用下列 put-key-policy 命令新增更新的政策：

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

步驟 3：為 KMS 金鑰與您的查詢結果建立關聯

若要為 KMS 金鑰與帳戶中的查詢結果建立關聯

使用 disassociate-kms-key 命令，如下所示：

```
aws logs associate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-result:*" --kms-key-id "key-arn"
```

步驟 4：將金鑰與帳戶中的查詢結果取消關聯

若要取消 KMS 金鑰與查詢結果的關聯，請使用以下 disassociate-kms-key 命令：

```
aws logs disassociate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-result:*"
```

從 CloudWatch Logs Insights 查詢結果產生自然語言摘要

分析日誌資料對於了解應用程式的行為至關重要，但解譯大量日誌項目可能很耗時。CloudWatch Logs Insights 現在提供自然語言摘要功能，可將複雜的查詢結果轉換為清晰、簡潔的摘要。此功能可協助您快速識別問題，並從日誌資料中取得可行的洞見。

運作方式

CloudWatch Logs Insights 可以使用 Amazon Bedrock 從查詢結果產生人類可讀取的摘要。此功能支援所有 CloudWatch Logs Insights 查詢語言，並提供清晰、可行的日誌資料洞見。

區域可用性和資料處理

> **Important**
>
> 當您使用此功能時，您的查詢結果可能會在不同的 中處理 AWS 區域。例如，如果您在美國東部 （維吉尼亞北部） 執行查詢，則摘要可能發生在美國西部 （奧勒岡）。

下表列出查詢結果功能可用 AWS 區域 之不同地理位置的可能處理方式：

支援的 CloudWatch Logs 地理位置	可能的處理區域
美國 (US)	美國東部 (維吉尼亞北部) 區域
	美國東部 (俄亥俄) 區域
	美國西部 (奧勒岡) 區域

支援的 CloudWatch Logs 地理位置	可能的處理區域
歐洲	歐洲 (法蘭克福) 區域 歐洲 (愛爾蘭) 區域 歐洲 (巴黎) 區域 歐洲 (斯德哥爾摩) 區域 歐洲 (倫敦) 區域
亞太區域	美國東部 (維吉尼亞北部) 區域 美國東部 (俄亥俄) 區域 美國西部 (奧勒岡) 區域
南美洲	美國東部 (維吉尼亞北部) 區域 美國東部 (俄亥俄) 區域 美國西部 (奧勒岡) 區域

開始使用

產生自然語言摘要

1. 執行 CloudWatch Logs Insights 查詢。
2. 查詢完成後，選取摘要結果。

許可

您必須具有下列其中一項：

- `CloudWatchLogsFullAccess` 許可
- `CloudWatchLogsReadOnlyAccess` 許可
- 自訂 IAM 政策，包括 `cloudwatch:GenerateQueryResultsSummary`、`logs:GetQueryResultslogs:DescribeQueries`和 `logs:FilterLogEvents`動作

資料隱私權

您的查詢結果會安全地處理，不會用於訓練或改善 CloudWatch Logs Insights 或 Amazon Bedrock。如果您選擇使用意見回饋按鈕提供查詢結果摘要的意見回饋，您的意見回饋會指出您對 CloudWatch Logs Insights 所提供功能的滿意度。

使用排程查詢自動化日誌分析

排程查詢可讓您定期自動執行 CloudWatch Logs Insights 查詢。您可以設定排程查詢自動執行，並將結果交付至目的地，例如 Amazon S3 儲存貯體或 Amazon EventBridge 事件匯流排，而不是手動執行查詢來分析日誌資料。此自動化非常適合產生定期報告、監控趨勢，或根據日誌分析結果觸發下游程序。

排程查詢支援 CloudWatch Logs Insights 中提供的所有三種查詢語言：

- Logs Insights 查詢語言 (Logs Insights QL)
- OpenSearch Service 管道處理語言 (PPL)
- OpenSearch Service 結構化查詢語言 (SQL)

目錄

了解排程查詢概念

在建立排程查詢之前，請了解這些會影響查詢執行方式和結果交付位置的關鍵概念。

IAM 角色分離

排程查詢需要兩個不同的 IAM 角色：一個用於執行查詢，另一個用於將結果交付至 Amazon S3 或 Amazon EventBridge 事件匯流排。了解此分隔存在的原因可協助您正確設定許可，並使用其提供的安全性和操作優勢。

雙角色架構會劃分資料存取和資料交付之間的責任。查詢執行角色會存取您的日誌資料並執行查詢，而目的地交付角色會將結果寫入您選擇的目的地。此區隔遵循最低權限原則 - 每個角色只有其特定函數所需的許可。

查詢執行角色

允許 CloudWatch Logs 代表您執行 CloudWatch Logs Insights 查詢。此角色需要存取日誌群組和執行查詢的許可，但不需要存取目的地資源。必要許可：

- logs:StartQuery
- logs:StopQuery
- logs:GetQueryResults
- logs:DescribeLogGroups
- logs:Unmask 如果需要取消遮罩資料

對於 KMS 加密的日誌群組：用於加密日誌群組的 KMS 金鑰的 kms:Decrypt和 kms:DescribeKey許可。也需要新增這些許可。

信任關係要求：查詢執行角色必須包含允許 CloudWatch Logs 服務 (logs.amazonaws.com) 擔任角色的信任政策。如果沒有此信任關係，排程的查詢將會失敗並出現許可錯誤。

查詢執行角色的信任政策範例：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

查詢執行角色的許可政策範例：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:StartQuery",
```

```
                "logs:StopQuery",
                "logs:GetQueryResults",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        }
    ]
}
```

目的地交付角色

允許 CloudWatch Logs 將查詢結果交付至您選擇的目的地。根據最低權限原則，此角色只需要特定目的地服務的許可。必要許可因目的地類型而異。

信任關係需求：目的地交付角色也必須包含信任政策，允許 CloudWatch Logs 服務 (logs.amazonaws.com) 擔任該角色。

S3 目的地交付角色的許可政策範例：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::your-scheduled-query-results-bucket/*"
        }
    ]
}
```

此區隔為您的操作提供實際的好處。從安全角度來看，如果您需要變更交付結果的位置，您只需修改目的地交付角色，而不變更查詢執行許可。為了合規和稽核，您可以清楚地追蹤哪些角色存取敏感日誌資料，以及哪些角色寫入外部系統。這可讓您更輕鬆地示範日誌分析基礎設施遵循安全最佳實務。

跨區域和跨帳戶用量

排程查詢會在特定區域中建立，並在該區域中執行。不過，您可以查詢日誌群組，並跨區域和帳戶交付結果。您需要將一或多個 AWS 帳戶設定為監控帳戶，並將其連結至多個來源帳戶。監控帳戶是中央 AWS 帳戶，可檢視來源帳戶產生的可觀測性資料並與之互動。來源帳戶是個別 AWS 帳戶，可產生其

中資源的可觀測性資料。來源帳戶會與監控帳戶共用其可觀察性資料。因此，您可以使用所有連結帳戶的日誌群組，從監控帳戶設定排程查詢。

查詢跨區域日誌群組

您的排程查詢可以存取任何區域中的日誌群組。使用完整的 ARN 格式指定日誌群組：`arn:aws:logs:region:account-id:log-group:log-group-name`。查詢執行角色需要`logs:StartQuery`和所有目標區域中日誌群組的`logs:GetQueryResults`許可。

Important

查詢日誌群組或跨區域交付結果時，日誌資料會跨越區域界限。考慮下列各項：

- 資料駐留要求 - 確保跨區域資料傳輸符合組織的資料控管政策和法規要求
- 資料傳輸成本 - 跨區域資料傳輸會產生額外費用
- 網路延遲 - 在遙遠區域中存取日誌群組的查詢可能會遇到較高的延遲

為了獲得最佳效能和成本效益，請在與主要日誌群組相同的區域中建立排程查詢。

替代方法：使用 CloudWatch Logs 集中，將多個帳戶和區域的日誌資料複寫至中央監控帳戶。這可讓您在存取所有集中式日誌的單一區域中建立排程查詢，避免跨區域查詢並簡化 IAM 許可管理。

排程表達式和時區處理

您定義的排程會決定查詢執行的時間及其執行頻率。選擇正確的排程表達式會影響您何時收到結果，以及您查詢的資料量。了解表達式類型可協助您在簡單性和精確度之間進行選擇。

Cron 表達式提供對計時的精確控制，可讓您指定確切的時間、星期幾或月份天數。當您需要在特定上班時間執行查詢或符合操作排程時，請使用 cron 表達式。在 主控台中，您也可以使用簡單的行事曆選項來排程查詢。

Cron 表達式

在特定時間執行查詢。格式：`cron(minute hour day-of-month month day-of-week year)`。範例：

- `cron(0 9 * * ? *)` - 每日上午 9：00 UTC
- `cron(0 18 ? * MON-FRI *)` - 工作日 UTC 下午 6：00

- `cron(0 0 1 * ? *)` - 每月第一天午夜 UTC
- `cron(0 12 ? * SUN *)` - 每週日中午 UTC
- `cron(30 8 1 1 ? *)` - 1 月 1 日上午 8：30 UTC

所有排程查詢都會在 UTC 中執行，無論您的本機時區或 AWS 資源所在的位置為何。當您排定工作時間或時間敏感分析的查詢時，這尤其重要。例如，如果您的業務在美國東部時間營運，而且您想要在東部時間上午 9 點進行每日報告，您需要考慮 UTC 偏移（夏令時間 14：00 UTC，否則為 13：00 UTC)。以 UTC 為考量來規劃排程表達式，以確保查詢在預期時間執行。

選擇查詢語言

排程查詢支援三種不同的查詢語言，而且您的選擇會影響您撰寫查詢的方式，以及您的團隊可以輕鬆維護查詢的方式。正確的語言取決於您的分析需求和您團隊的現有技能。

如果您主要是篩選和彙總日誌資料，CloudWatch Logs Insights 查詢語言提供最直接的語法。對於您需要透過多個步驟重塑或豐富資料的複雜資料轉換，PL 的管道方法可讓您更輕鬆地遵循邏輯。當您需要執行與資料庫操作類似的聯結或複雜彙總時，SQL 會提供熟悉的語法，讓資料庫經驗豐富的團隊可以快速採用。

CloudWatch Logs Insights 查詢語言 (CWLI)

專為使用直覺式語法進行日誌分析而打造。最適合：

- 文字型日誌分析和篩選
- 時間序列彙總和統計資料
- 記錄分析的新團隊

OpenSearch Service 管道處理語言 (PPL)

具有強大資料轉換功能的管道型查詢語言。最適合：

- 複雜的資料轉換和擴充
- 多步驟資料處理工作流程
- 熟悉管道型處理的團隊

OpenSearch Service 結構化查詢語言 (SQL)

熟悉資料庫樣式查詢的標準 SQL 語法。最適合：

- 複雜聯結和彙總
- 商業智慧和報告

- 具有強大 SQL 體驗的團隊

目的地選擇和使用案例

您傳送查詢結果的位置會決定您可以如何處理它們。無論您要建置長期分析、觸發自動回應或兩者，此選項都會塑造您的整個下游工作流程。了解每個目的地類型的強項，可協助您為使用案例設計正確的架構。

Amazon S3 目的地已針對儲存和批次處理進行最佳化。當您需要將查詢結果保留數月或數年、分析一段時間的趨勢，或將資料饋送至分析平台時，Amazon S3 會提供具有無限制保留的符合成本效益的儲存體。EventBridge 目的地已針對即時自動化進行最佳化。當查詢結果應觸發立即動作時，例如傳送提醒、啟動工作流程或更新系統時，EventBridge 會以您的應用程式可立即回應的事件的形式提供結果。根據預設，所有查詢完成事件都會自動做為事件傳送至預設事件匯流排，以便與下游處理系統、Lambda 函數或其他事件驅動型架構整合。結果只會在成功執行查詢時發佈到目的地。

Amazon S3 目的地

將查詢結果儲存為 JSON 檔案，以進行長期保留和批次處理。最適合：

- 歷史分析和資料封存
- 與資料湖和分析平台整合
- 合規和稽核要求
- 經濟實惠的大型結果集儲存

EventBridge 目的地

將查詢結果做為事件傳送，以進行即時處理和自動化。您只能使用事件中傳送的 queryId 擷取查詢結果，最多 30 天，因為我們將結果存放 30 天。最適合：

- 觸發自動回應以查詢結果
- 與無伺服器工作流程和 Lambda 函數整合
- 即時提醒和通知系統
- 事件驅動型架構和微服務

查詢結果格式和結構

對於 Amazon S3 目的地 - 查詢結果會以 JSON 格式交付，其結構與 GetQueryResults API 回應相同。為了讓 Amazon EventBridge 了解排程查詢結果的格式，可協助您設計下游處理和整合工作流程。

查詢結果會以 JSON 格式交付，結構如下：

```
{
    "version": "0",
    "id": "be72061b-eca2-e068-a7e1-83e01d6fe807",
    "detail-type": "Scheduled Query Completed",
    "source": "aws.logs",
    "account": "123456789012",
    "time": "2025-11-18T11:31:48Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:logs:us-east-1:123456789012:scheduled-query:477b4380-b098-474e-9c5e-
e10a8cc2e6e7"
    ],
    "detail": {
        "queryId": "2038fd57-ab4f-4018-bb2f-61d363f4a004",
        "queryString": "fields @timestamp, @message, @logStream, @log\n| sort
 @timestamp desc\n| limit 10000",
        "logGroupIdentifiers": [
            "/aws/lambda/my-function"
        ],
        "status": "Complete",
        "startTime": 1763465460,
        "statistics": {
            "recordsMatched": 0,
            "recordsScanned": 0,
            "estimatedRecordsSkipped": 0,
            "bytesScanned": 0,
            "estimatedBytesSkipped": 0,
            "logGroupsScanned": 1
        }
    }
}
```

關鍵元素包括：

- `statistics` - 查詢效能指標，包括相符、掃描、處理位元組和估計略過資料的記錄
- `startTime` - 查詢執行開始的時間 (Unix 時間戳記)
- `queryString` - 實際執行的查詢
- `queryId` - 查詢的查詢 ID，可使用其擷取結果
- `logGroupIdentifiers` - 查詢的日誌群組清單

- `status` - 查詢執行狀態（完成、失敗等）

排程表達式參考

使用這些參考表來建構排程查詢的排程表達式。所有時間均以 UTC 表示。

Cron 表達式語法

格式：`cron(minute hour day-of-month month day-of-week year)`

使用案例	Cron 表達式	Description	使用時機
每日排程	`cron(0 9 * * ? *)`	UTC 每天上午 9：00	每日報告
	`cron(0 */6 * * ? *)`	每 6 小時 (00：00、06：00、12：00、18：00 UTC)	頻繁監控
	`cron(30 2 * * ? *)`	UTC 每天上午 2：30	離峰分析
營業時間	`cron(0 9-17 ? * MON-FRI *)`	週一至週五 UTC，上午 9 點至下午 5 點，每小時	業務監控
	`cron(0 18 ? * MON-FRI *)`	UTC 的工作日下午 6：00	工作日結束
	`cron(0 8,12,17 ? * MON-FRI *)`	UTC 工作日上午 8 點、中午和下午 5 點	關鍵營業時間
每週排程	`cron(0 12 ? * SUN *)`	每週日中午 UTC	每週摘要
	`cron(0 9 ? * MON *)`	UTC 每週一上午 9：00	週開始報告
	`cron(0 23 ? * FRI *)`	每週五 11：00 PM UTC	週結束清除
每月排程	`cron(0 0 1 * ? *)`	每月第一天午夜 UTC	每月報告
	`cron(0 9 L * ? *)`	每月最後一天 UTC 上午 9：00	月末處理

使用案例	Cron 表達式	Description	使用時機
	`cron(0 10 1 1,4,7,10 ? *)`	每季第一天 UTC 上午 10：00	每季分析
高頻率	`cron(*/15 * * * ? *)`	每 15 分鐘	即時監控
	`cron(0,30 * * * ? *)`	每 30 分鐘（在 ：00 和 ：30)	頻繁檢查
	`cron(0 */2 * * ? *)`	每 2 小時	定期間隔
特殊案例	`cron(30 8 1 1 ? *)`	1 月 1 日上午 8：30 UTC	年度報告
	`cron(0 6 * * SAT,SUN *)`	UTC 上午 6：00 的週末	週末處理
	`cron(0 0 ? * MON#1 *)`	每月的第一個星期一午夜 UTC	每月規劃

Cron 表達式欄位參考

欄位	值	萬用字元	範例
分鐘（第 1 個）	0-59	`* , - /`	`0`（小時頂端）、 `*/15`（每 15 分鐘）、 `0,30`（每小時兩次）
小時（第 2 個）	0-23	`* , - /`	`9`（上午 9 點）、 `*/2`（每 2 小時）、 `9-17`（上班時間）
Day-of-month（第 3 天）	1-31、L、W	`* , - / ?`	`1`（第一天）、 `L`（最後一天）、 `?`（使用day-of-week時）
月（第 4 個）	1-12 或 JAN-DEC	`* , - /`	`1`(1 月）`JAN`、、 `1,4,7,10`（每季）
Day-of-week（第 5 天）	1-7 或 SUN-SAT	`* , - / ? # L`	`MON-FRI`（平日）`SUN`、、 `MON#1`（第一個星期一）
年（第 6 個）	1970-2199	`* , - /`	`*`（每年）、 `2024`（特定年份）、 `2024-2026`（範圍）

萬用字元和特殊表達式

***** (星號)

符合 欄位中的所有值。範例：*在小時欄位中表示每小時。

?（問號）

沒有特定值。在指定其他 時，以day-of-month或day-of-week使用 。範例：?在day-of-month時MON-FRI，在每月day-of-week。

- (破折號)

值的範圍。範例： MON-FRI （週一至週五）、 9-17（上午 9 點至下午 5 點）。

,（逗號）

多個特定值。範例： MON,WED,FRI （週一、週三、週五）、 8,12,17（上午 8 點、中午、下午 5 點）。

/（斜線）

步驟值或增量。範例：0/15以分鐘為單位表示從分鐘 0 (0、15、30、45) 開始每 15 分鐘一次。以小時*/2為單位表示每 2 小時一次。

L（最後一個）

每月的最後一天或工作日的最後一次出現。範例：以月中的L日表示月中的最後一天。 FRIL表示月中的最後一個星期五。 day-of-month

W（工作日）

最近的工作日。範例： 15W 表示最接近當月 15 日的工作日。

#（第 n 次出現）

每月第三次出現工作日。範例： MON#1 表示每月的第一個星期一， FRI#2表示每月的第二個星期五。

常見模式和最佳實務

- 對於商業應用程式：使用 MON-FRI和 上班時間 （例如 9-17)，以避免在週末或下班時間執行查詢。
- 對於高頻率監控：使用增量，例如 */15（每 15 分鐘），但請注意查詢並行限制。

- 為了提高資源效率：使用 2-6 UTC 等早間時間，在離峰時間排程資源密集型查詢。
- 對於每月報告：將 L 用於月份的最後一天或特定日期，例如1第一天，以確保一致的時間。

最佳實務

遵循這些最佳實務，以確保可靠且有效率的排程查詢操作：

查詢最佳化

- 在排程之前手動測試查詢，以驗證效能和結果
- 在查詢的早期使用篩選條件索引來減少資料處理
- 限制時間範圍，以避免大量日誌群組逾時
- 考慮查詢複雜性和執行時間限制

排程規劃

- 確保查詢在下一次排程執行之前完成，以避免重疊執行
- 設定時間範圍時，請考慮日誌擷取延遲
- 在特定時間使用 cron 表達式
- 分散排程，以確保您不會達到查詢並行限制

監控和維護

- 定期監控執行歷史記錄，以識別故障或效能問題
- 定期檢閱和更新 IAM 角色以維護安全性
- 在部署到生產環境之前測試目的地可存取性

Authorization

- 排程查詢的所有 APIs 都會在排程查詢資源上進行授權，而不是在日誌群組等輸入中採用的資源上進行授權。相應地設定 IAM 政策
- 使用 APIs 中傳遞的執行角色管理日誌群組的授權

排程查詢入門

建立排程查詢時，您將設定數個關鍵元件，以定義查詢的執行方式以及交付結果的位置。了解這些元件可協助您設定有效的自動化日誌分析。

每個排程查詢都包含下列關鍵元件：

查詢組態

用於分析的 CloudWatch Logs Insights 查詢字串、目標日誌群組和查詢語言。

排程表達式

定義查詢執行時間的 Cron 表達式或頻率行事曆。您可以指定時區設定，以確保查詢在正確的本機時間執行。主控台會顯示排程的人類可讀描述，例如「在每個星期二的 15：10 執行查詢，時間範圍為 5 分鐘，立即在 UTC 上生效，直到無限期為止。」

時間範圍

每個查詢執行的回顧期間，由與執行時間偏移的開始時間所定義。這會決定每個查詢執行將分析多少歷史資料。

執行排程預覽

主控台會顯示接下來的三個排程查詢執行，其中包含確切的日期和時間 （例如 2025/10/28 15：10、UTC；2025/11/04 15：10、UTC；2025/11/11 15：10、UTC)，可協助您確認排程已正確設定。

目的地

查詢結果在成功執行後交付的位置。支援的目的地包括 Amazon S3 儲存貯體，預設會將結果中繼資料傳送至預設事件匯流排。

執行角色

CloudWatch Logs 擔任的 IAM 角色，可執行查詢並將結果交付至指定的目的地。

建立排程查詢之前，請確定您已設定必要的許可和資源。

建立排程查詢

建立排程查詢，以自動執行 CloudWatch Logs Insights 查詢，並將結果交付至您選擇的目的地。

先決條件

建立排程查詢之前，請確定您有下列項目：

- 日誌群組 - 一或多個日誌群組，其中包含您要分析的資料
- 執行 IAM 角色 - 具有下列許可的 IAM 角色：
  - `logs:StartQuery` - 啟動 CloudWatch Logs Insights 查詢的許可

  - `logs:GetQueryResults` - 擷取查詢結果的許可
  - `logs:DescribeLogGroups` - 存取日誌群組資訊的許可。這僅適用於日誌群組探索的字首型日誌群組
- 目的地許可 - 所選目的地的其他 IAM 許可：
  - 對於 Amazon S3 目的地： `s3:PutObject`
- 對於 AWS CLI 和 API 使用 - 具有呼叫 CloudWatch Logs APIs許可的已設定 AWS 登入資料

如需詳細的 IAM 政策範例，請參閱 適用於 Amazon CloudWatch Logs 的 Identity and Access Management。另請注意，每個帳戶只能有 1000 個排程查詢。

Console

建立排程查詢 （主控台）

1. 在 https：// 開啟 CloudWatch Logs 主控台？ https://us-east-1.console.aws.amazon.com/cloudwatch/homeregion=us-east-1#logsV2：logs-insights。
2. 在導覽窗格中，選擇 Logs Insights。
3. 選擇建立排程查詢。
4. 在查詢定義區段中：
   a. 針對查詢語言，從清單中選擇要使用的查詢語言。
   b. 針對查詢字串，在方塊中輸入您的 CloudWatch Logs Insights 查詢。
   c. 對於日誌群組，從清單中選擇要查詢的日誌群組。
5. 在排程設定區段中：
   a. 對於排程表達式，請在查詢執行時設定 。從預先定義的選項中選擇或輸入自訂 Cron 表達式。
   b. 對於建立時生效，請指定排程何時變成作用中。選擇立即開始，或使用 YYYY/MM/DD 格式在特定日期和時間開始。
   c. 針對時間範圍，指定每個查詢執行的回顧期間。以分鐘為單位輸入持續時間，以定義從執行時間開始查詢的時間。
   d. 對於無限期繼續，請指定排程何時結束。選擇無限期執行，或使用 YYYY/MM/DD 格式直到特定日期和時間。
6. 主控台會根據您的組態顯示接下來三個排程的查詢執行，以 UTC 顯示執行查詢的確切日期和時間。

7. 在將查詢結果發佈至 S3 - 選用區段中（如果使用 S3 目的地）：

   a. 對於 S3 儲存貯體，如果目的地儲存貯體位於相同 AWS 帳戶，請選取此帳戶；如果儲存貯體位於不同 AWS 帳戶，請選取另一個帳戶，並提供儲存貯體擁有帳戶的帳戶 ID 做為輸入。

   b. 對於 Amazon S3 URI，輸入將存放結果的 Amazon S3 儲存貯體和字首（例如 `s3://my-bucket/query-results/`)。如果您選取此帳戶，您可以選擇瀏覽 Amazon S3 以導覽並選取現有的 Amazon S3 位置。

   c. （選用）對於 KMS 金鑰 ARN，輸入客戶受管 AWS KMS 金鑰的 ARN，以使用 SSE-KMS 加密查詢結果。金鑰必須與目的地 Amazon S3 儲存貯體位於相同的 AWS 區域。

8. 在用於將查詢結果發佈至 Amazon S3 的 IAM 角色區段中，選擇下列其中一個選項：

   a. 選擇自動建立具有預設許可的新角色，以自動設定具有 CloudWatch Logs 將查詢結果交付至 Amazon S3 所需許可的 IAM 角色。

   b. 選擇使用現有角色來選取具有所需政策的現有 IAM 角色，以便 CloudWatch Logs 將查詢結果交付至 Amazon S3。使用搜尋欄位，從清單中選擇適當的 IAM 角色。

9. 在排程查詢執行的 IAM 角色區段中，選擇下列其中一個選項：

   a. 選擇自動建立具有預設許可的新角色，以自動設定具有 CloudWatch Logs 執行排程查詢所需許可的 IAM 角色。

   b. 選擇使用現有角色來選取具有所需政策的現有 IAM 角色，以便 CloudWatch Logs 執行排程查詢。使用搜尋欄位，從清單中選擇適當的 IAM 角色。

10. 選擇建立排程以建立排程查詢。

AWS CLI

建立排程查詢 (AWS CLI)

- 使用 `create-scheduled-query`命令建立新的排程查詢：

```
aws logs create-scheduled-query \
    --name "ErrorAnalysisQuery" \
    --query-language "CWLI" \
    --query-string "fields @timestamp, @message | filter @message like /ERROR/ |
 stats count() by bin(5m)" \
    --schedule-expression "cron(8 * * * ? *)" \
    --execution-role-arn "arn:aws:iam::123456789012:role/
CloudWatchLogsScheduledQueryRole" \
```

```
    --log-group-identifiers "/aws/lambda/my-function" "/aws/apigateway/my-api" \
    --state "ENABLED"
```

API

建立排程查詢 (API)

- 使用 CreateScheduledQuery動作建立新的排程查詢。下列範例會建立排程查詢，每小時執行一次：

```
{
    "name": "ErrorAnalysisQuery",
    "queryLanguage": "CWLI",
    "queryString": "fields @timestamp, @message | filter @message like /ERROR/ |
 stats count() by bin(5m)",
    "scheduleExpression": "cron(8 * * * ? *)",
    "executionRoleArn": "arn:aws:iam::123456789012:role/
CloudWatchLogsScheduledQueryRole",
    "logGroupIdentifiers": ["/aws/lambda/my-function", "/aws/apigateway/my-
api"],
    "state": "ENABLED"
}
```

建立排程查詢後，您可以從排程查詢頁面並使用 ListScheduledQueries API 來檢視和管理它，該 API 會顯示所有排程查詢及其名稱、建立日期、上次執行狀態、上次觸發時間和重複頻率。

檢視和管理排程查詢

下列資訊適用於每個查詢：

名稱

您指派給排程查詢的唯一名稱。選取名稱以檢視詳細的組態和執行歷史記錄。

Creation (建立) 日期

建立排程查詢的日期，以 YYYY-MM-DD 格式顯示。

上次執行的狀態

最近查詢執行的執行狀態。可能的值包括：

- 完成 - 已成功執行查詢，並將結果交付至所有設定的目的地。
- 失敗 - 查詢執行或結果交付失敗。檢查執行歷史記錄以取得錯誤詳細資訊。
- 無效查詢 - 查詢無效且具有語法問題
- 逾時 - 查詢已逾時。查詢會在 60 分鐘後自動逾時

上次觸發時間

上次執行查詢的日期和時間，以 YYYY-MM-DD HH：MM：SS 格式顯示。如果查詢尚未執行，則會顯示永不。

重複每個

查詢的排程頻率。針對使用 Cron 表達式的查詢或更簡單排程的特定頻率描述，顯示自訂。

排程查詢頁面提供所有排程查詢的概觀，顯示其目前狀態和執行歷史記錄，以便您可以從集中位置檢視、監控和管理所有排程查詢。使用此資訊來監控查詢效能、識別問題，以及管理您的自動化日誌分析工作流程。

Console

檢視排定的查詢 （ 主控台 ）

1. 在 https：// 開啟 CloudWatch Logs 主控台？ https://us-east-1.console.aws.amazon.com/cloudwatch/homeregion=us-east-1#logsV2：logs-insights。
2. 在 CloudWatch Logs 主控台中，選擇排程查詢、檢視排程查詢。

AWS CLI

列出排程查詢 (AWS CLI)

- 使用 `list-scheduled-queries`命令列出所有排定的查詢：

```
aws logs list-scheduled-queries --max-results 10
```

API

列出排程查詢 (API)

- 使用 `ListScheduledQueries`動作來擷取所有排定的查詢：

```
{
    "maxResults": 10
}
```

已排程查詢頁面標頭會顯示您帳戶中已排程查詢的總數，協助您追蹤用量並有效管理自動化日誌分析工作流程。

檢視排定的查詢執行歷史記錄

使用執行歷史記錄來監控排程查詢的效能，並針對查詢執行或結果交付的任何問題進行疑難排解。

執行歷史記錄會顯示每個查詢執行的狀態，包括成功執行、失敗和目的地處理結果。您可以使用此資訊來識別模式、診斷問題，並確認查詢是否如預期般執行。

Console

檢視執行歷史記錄 （主控台）

1. 在 CloudWatch Logs 主控台中，選擇排程查詢、檢視排程查詢。
2. 選取您要檢查的排程查詢。
3. 選擇 Execution history (執行歷程記錄) 標記。

AWS CLI

檢視執行歷史記錄 (AWS CLI)

1. 使用 `get-scheduled-query-history`命令來擷取排程查詢的執行歷史記錄：

```
aws logs get-scheduled-query-history \
    --identifier "DailyErrorMonitoring" \
    --start-time 1743379200 \
    --end-time 1743465600 \
    --max-results 10
```

2. 若要依執行狀態篩選，請新增 `--execution-statuses` 參數：

```
aws logs get-scheduled-query-history \
    --identifier "DailyErrorMonitoring" \
    --start-time 1743379200 \
```

```
    --end-time 1743465600 \
    --max-results 1 \
    --execution-statuses "SUCCEEDED"
```

API

檢視執行歷史記錄 (API)

- 使用 GetScheduledQueryHistory動作來擷取執行歷史記錄：

```
{
    "identifier": "DailyErrorMonitoring",
    "startTime": 1743379200,
    "endTime": 1743465600,
    "maxResults": 10,
    "executionStatuses": ["SUCCEEDED", "FAILED"]
}
```

執行歷史記錄會顯示：

- 執行狀態 - 執行中、完成、失敗、逾時或 InvalidQuery
- 觸發時間 - 執行查詢的時間
- 目的地 - 每個已設定目的地的處理狀態，包括 S3 和 EventBridge
- 錯誤訊息 - 有關查詢執行或目的地處理中任何失敗的詳細資訊

更新排程查詢

修改您的排程查詢組態，以隨著需求的變化變更查詢字串、排程、目的地或執行角色。

您可以更新排程查詢的任何方面，包括查詢字串、排程表達式、目的地和執行角色。對於未來的執行，變更會立即生效。

Console

更新排程查詢 （主控台）

1. 在 CloudWatch Logs 主控台中，選擇排程查詢、檢視排程查詢。
2. 選取您要更新的排程查詢。

3. 選擇編輯。
4. 視需要修改組態。
5. 選擇儲存變更。

AWS CLI

更新排程查詢 (AWS CLI)

- 使用 update-scheduled-query命令來修改現有的排程查詢：

```
aws logs update-scheduled-query \
    --identifier "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f" \
    --description "Monitor for ERROR level logs daily" \
    --query-language "LogsQL" \
    --query-string "fields @timestamp, @message | filter @message like /ERROR/"
 \
    --log-group-identifiers "/aws/lambda/my-function-1" "/aws/lambda/my-
function-2"
```

API

更新排程查詢 (API)

1. 使用 UpdateScheduledQuery動作來修改排程查詢組態：

```
{
    "identifier": "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f",
    "queryString": "fields @timestamp, @message | filter @message like /WARNING|
ERROR/ | stats count() by bin(5m)",
    "scheduleExpression": "cron(0 */2 * * ? *)",
    "state": "ENABLED"
}
```

2. 若要一次更新多個組態參數：

```
{
    "identifier": "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f",
```

```
    "queryString": "fields @timestamp, @message, @level | filter @level =
 'ERROR'",
    "scheduleExpression": "cron(0 8,12,16 * * ? *)",
    "executionRoleArn": "arn:aws:iam::111122223333:role/
UpdatedScheduledQueryRole",
    "logGroupIdentifiers": ["/aws/lambda/my-function", "/aws/lambda/another-
function"],
    "destinationConfiguration": {
        "s3Configuration": {
            "destinationIdentifier": "s3://111122223333-sqn-results-bucket/
processed-results",
            "roleArn": "arn:aws:iam::111122223333:role/Admin"
        }
    }
}
```

設定排程查詢的 S3 目的地

將 Amazon S3 設定為目的地，將排程查詢結果儲存為 JSON 檔案，以進行長期保留和分析。

使用 Amazon S3 做為目的地時，查詢結果會以 JSON 檔案的形式儲存在指定的儲存貯體和字首中。此選項非常適合封存結果、執行批次分析，或與處理 S3 資料的其他 AWS 服務整合。

您可以將查詢結果交付至與排程查詢相同的 AWS 帳戶中的 Amazon S3 儲存貯體，或交付至不同 AWS 帳戶中的儲存貯體。您也可以選擇使用客戶受管 AWS KMS 金鑰 (SSE-KMS) 加密查詢結果。

將結果交付至相同帳戶中的 Amazon S3 儲存貯體

當目的地 Amazon S3 儲存貯體與排程查詢位於相同 AWS 帳戶時，您可以直接從主控台瀏覽並選取儲存貯體。

設定相同帳戶的 Amazon S3 目的地（主控台）

1. 在將查詢結果發佈至 S3 區段中，針對 S3 儲存貯體，選取此帳戶。
2. 針對 Amazon S3 URI，輸入將存放結果的 Amazon S3 儲存貯體和字首（例如 `s3://my-bucket/query-results/`），或選擇瀏覽 Amazon S3 以導覽並選取現有的 Amazon S3 位置。
3. （選用）若要使用客戶受管 AWS KMS 金鑰加密結果，請在 AWS KMS KMS 金鑰 ARN 欄位中輸入金鑰的 ARN。金鑰必須與目的地 Amazon S3 儲存貯體位於相同的 AWS 區域。如果您未指定 AWS KMS 金鑰，則會套用儲存貯體的預設加密設定。

4. 在用於將查詢結果發佈至 Amazon S3 的 IAM 角色區段中，選擇使用預設許可自動建立新角色以自動設定所需許可，或選擇使用現有角色來選取具有所需政策的現有 IAM 角色。

目的地交付 IAM 角色需要下列許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

將結果交付至另一個帳戶中的 Amazon S3 儲存貯體

您可以將排定的查詢結果交付至不同 AWS 帳戶中的 Amazon S3 儲存貯體。使用跨帳戶儲存貯體時，您必須提供 Amazon S3 URI 和儲存貯體擁有帳戶的帳戶 ID。

設定跨帳戶 Amazon S3 目的地（主控台）

1. 在將查詢結果發佈至 S3 區段中，針對 S3 儲存貯體，選取另一個帳戶，並提供儲存貯體擁有帳戶的帳戶 ID 做為輸入。
2. 對於 Amazon S3 URI，輸入另一個帳戶中目的地儲存貯體和字首的完整 Amazon S3 URI（例如 `s3://cross-account-bucket/query-results/`）。
3. （選用）若要使用客戶受管 AWS KMS 金鑰加密結果，請在 AWS KMS KMS 金鑰 ARN 欄位中輸入金鑰的 ARN。金鑰必須與目的地 Amazon S3 儲存貯體位於相同的 AWS 區域。
4. 在用於將查詢結果發佈至 Amazon S3 的 IAM 角色區段中，選擇使用預設許可自動建立新角色以自動設定所需許可，或選擇使用現有角色來選取具有所需政策的現有 IAM 角色。

跨帳戶交付需要雙方的許可。來源帳戶中的目的地交付 IAM 角色需要下列許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::cross-account-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

目的地帳戶中的 Amazon S3 儲存貯體政策必須授予來源帳戶的 IAM 角色寫入物件的許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowScheduledQueryRolePutObject",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/my-s3-delivery-role"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::cross-account-bucket/prefix/*"
        }
    ]
}
```

使用客戶受管 AWS KMS 金鑰加密結果

您可以選擇性地指定客戶受管 AWS KMS 金鑰，以使用 SSE-KMS 加密交付至 Amazon S3 的查詢結果。 AWS KMS 金鑰可以位於與排程查詢相同的帳戶中，也可以位於不同的帳戶中。

當您指定 AWS KMS 金鑰時，排程查詢會使用該金鑰透過 SSE-KMS 加密結果。當您未指定 AWS KMS 金鑰時，會套用儲存貯體的預設加密設定。如果使用客戶受管金鑰以預設 SSE-KMS 加密設定儲存貯體，則目的地交付 IAM 角色仍必須擁有該金鑰的`kms:GenerateDataKey`許可。

目的地交付 IAM 角色需要 AWS KMS 金鑰的kms:GenerateDataKey許可。下列範例顯示具有客戶受管 AWS KMS 金鑰的 Amazon S3 目的地所需的許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "111122223333"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey",
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.us-east-1.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::my-bucket*"
                }
            }
        }
    ]
}
```

當 AWS KMS 金鑰位於與目的地交付 IAM 角色不同的帳戶中時，擁有金鑰帳戶中的 AWS KMS 金鑰政策必須明確授予角色存取權：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowScheduledQueryRoleToEncrypt",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/my-s3-delivery-role"
```

```
            },
            "Action": "kms:GenerateDataKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "s3.us-east-1.amazonaws.com"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::my-bucket*"
                }
            }
        }
    ]
}
```

> **Note**
>
> 當 AWS KMS 金鑰和目的地交付 IAM 角色位於相同帳戶時，如果 AWS KMS 金鑰政策包含預設的「啟用 IAM 政策」根陳述式，則單獨 IAM 身分政策就已足夠。只有在 AWS KMS 金鑰政策未委派給 IAM 時，才需要明確的金鑰政策授予。

用於將查詢結果發佈到 Amazon S3 的 IAM 角色必須與 IAM 角色分開設定，以便排程查詢執行。此區隔允許精細存取控制，其中執行角色可以執行查詢，而 Amazon S3 角色專門處理結果交付。這兩個角色都必須包含信任政策，允許 CloudWatch Logs 服務 (`logs.amazonaws.com`) 擔任該角色。

排定查詢的故障診斷

使用這些疑難排解主題來解決排程查詢的常見問題。

查詢執行失敗並出現許可錯誤

解決防止排程查詢執行或交付結果到目的地的許可錯誤。

當執行角色缺少從日誌群組讀取或寫入目的地資源的必要許可時，就會發生許可錯誤。

解決許可錯誤

1. 確認執行角色具有目標日誌群組的 `logs:GetQueryResults`、 `logs:StartQuery`和 `logs:DescribeLogGroups`許可。

2. 確定執行角色具有目的地資源的寫入許可（例如 s3:PutObject S3 儲存貯體的）。
3. 檢查信任政策是否允許 CloudWatch Logs 擔任執行角色。角色應該信任其信任政策中的日誌服務主體 (logs.amazonaws.com)。

常見原因包括缺少 IAM 許可、政策中的資源 ARNs 不正確，或信任政策組態問題。

若要防止許可錯誤，請在建立執行角色和測試許可時，使用最低權限原則，再將排程查詢部署至生產環境。

查詢逾時

解決排程查詢超過最大執行時間限制時發生的逾時錯誤。

當查詢需要超過 60 分鐘來處理指定的資料範圍時，就會發生查詢逾時，通常是因為大型資料集或複雜的查詢邏輯。

解決逾時錯誤

1. 透過減少開始時間偏移來減少每次執行的資料，以減少時間範圍。
2. 在查詢的早期新增篩選條件以最佳化查詢，以減少處理的資料量。使用篩選條件索引來減少資料掃描大小。
3. 考慮將複雜的查詢分成更簡單、更專注的查詢。

常見原因包括查詢大量時間範圍、處理大量日誌群組，或使用複雜的彙總而不進行適當的篩選。

為了防止逾時，請在 CloudWatch Logs Insights 中使用預期的資料磁碟區手動測試查詢，並在排程之前最佳化效能。

目的地處理失敗

解決排程查詢結果無法交付至設定目的地時發生的失敗。

當目標 Amazon S3 儲存貯體或 EventBridge 事件匯流排無法存取或設定不正確時，會發生目的地處理失敗。

解決查詢結果未發佈到目的地的失敗

1. 確認指定的 Amazon S3 儲存貯體存在且可存取。
2. 檢查目的地組態是否有正確的 URIs。

3. 確保執行角色具有寫入目的地的必要許可。

常見原因包括已刪除或重新命名目的地資源、不正確URIs 或網路連線問題。

為了防止目的地失敗，請定期驗證目的地組態並監控目的地資源可用性。

無效的查詢錯誤

解決排程查詢字串中無法成功執行的語法和邏輯錯誤。

當查詢字串包含語法錯誤、參考不存在的欄位，或使用不支援的查詢語言功能時，會發生無效的查詢錯誤。

解決無效的查詢錯誤

1. 在 CloudWatch Logs Insights 中手動測試查詢，以驗證語法和邏輯。
2. 檢查目標日誌群組中是否存在所有參考的日誌欄位。
3. 確認排程查詢支援您使用的查詢語言功能。

常見原因包括欄位名稱中的錯別字、查詢語法不正確，或使用排程執行環境中不支援的查詢功能。

為了防止無效的查詢錯誤，請務必在排程之前以互動方式測試查詢，並使用欄位探索功能來驗證欄位名稱。

查詢並行錯誤

當將並行錯誤視為排程查詢使用與 Cloudwatch Logs 洞見查詢相同的配額時，請注意以下幾點。建議您分散排程，以避免達到並行限制。

- 配額：每個 AWS 帳戶最多可以同時執行 100 個 CloudWatch Logs Insights 查詢。
- 儀表板：新增至 CloudWatch 儀表板的查詢也會計入此並行限制，因為它們會在載入或重新整理儀表板時執行。
- OpenSearch Service PPL/SQL：每個 AWS 帳戶最多可以同時執行 15 個 OpenSearch PPL 或 OpenSearch SQL 查詢。
- 跨帳戶查詢：並行配額適用於單一和跨帳戶查詢。使用 CloudWatch 跨帳戶可觀測性時，監控帳戶中針對連結來源帳戶啟動的查詢也會計入監控帳戶的並行限制。
- 不常存取日誌群組：對於不常存取日誌類別中的日誌群組，並行 Logs Insights 查詢的數量上限為 5 個。

日誌異常偵測

您可以透過兩種方式偵測日誌資料中的異常：建立日誌異常偵測器以進行持續監控，或使用 CloudWatch Logs Insights 查詢中的 `anomaly detection`命令進行隨需分析。

日誌異常偵測器會掃描擷取至日誌群組的日誌事件，並在日誌資料中自動尋找異常。異常偵測使用機器學習和模式辨識來建立典型日誌內容的基準。對於隨需分析，您可以在 CloudWatch Logs Insights 查詢中使用 `anomaly detection`命令來識別時間序列資料中的異常模式。如需查詢型異常偵測的詳細資訊，請參閱 在 CloudWatch Logs Insights 中使用異常偵測。

為日誌群組建立異常偵測器後，它會使用日誌群組中過去兩週的日誌事件進行訓練。訓練期間最多可能需要 15 分鐘。訓練完成後，它會開始分析傳入的日誌以識別異常，而異常會顯示在 CloudWatch Logs 主控台中供您檢查。

CloudWatch Logs 模式辨識透過識別日誌中的靜態和動態內容來擷取日誌模式。模式有助於分析大型日誌集，因為大量日誌事件通常可以壓縮為幾個模式。

例如，請參閱下列三個日誌事件的範例。

```
2023-01-01 19:00:01 [INFO] Calling DynamoDB to store for ResourceID: 12342342k124-12345
2023-01-01 19:00:02 [INFO] Calling DynamoDB to store for ResourceID: 324892398123-1234R
2023-01-01 19:00:03 [INFO] Calling DynamoDB to store for ResourceID: 3ff231242342-12345
```

在先前的範例中，這三個日誌事件都遵循一個模式：

```
<Date-1> <Time-2> [INFO] Calling DynamoDB to store for resource id <ResourceID-3>
```

模式內的欄位稱為字符。在模式內不同的欄位，例如請求 ID 或時間戳記，稱為動態字符。針對動態字符找到的每個不同值稱為字符值。

如果 CloudWatch Logs 可以推斷動態字符代表的資料類型，則會將字符顯示為 <*string*-*number*>。*##*是字符所代表資料類型的描述。相較於其他動態字符，*##*會顯示此字符在模式中的顯示位置。

CloudWatch Logs 會根據分析包含該名稱的日誌事件內容來指派名稱的字串部分。

如果 CloudWatch Logs 無法推斷動態權杖所代表的資料類型，它會將權杖顯示為 <Token-*number*>，且*##*指出與其他動態權杖相比，此權杖在模式中出現的位置。

動態字符的常見範例包括錯誤代碼、IP 地址、時間戳記和請求 IDs。

日誌異常偵測使用這些模式來尋找異常。在異常偵測器模型訓練期間之後，會根據已知趨勢評估日誌。異常偵測器會將顯著波動標記為異常。

本章說明如何啟用異常偵測、檢視異常、建立日誌異常偵測器的警示，以及日誌異常偵測器發佈的指標。它還描述了如何使用 加密異常偵測器及其結果 AWS Key Management Service。

建立日誌異常偵測器不會產生費用。

異常和模式的嚴重性和優先順序

日誌異常偵測器找到的每個異常都會指派優先順序。每個找到的模式都會指派一個嚴重性。

- 優先順序會自動計算，並以模式的嚴重性等級和與預期值的偏差量為基礎。例如，如果某個字符值突然增加 500%，該異常可能指定為HIGH優先順序，即使其嚴重性為 NONE。
- 嚴重性僅基於模式中找到的關鍵字，例如 FATAL、 ERROR和 WARN。如果找不到這些關鍵字，則模式的嚴重性會標記為 NONE。

異常可見性時間

當您建立異常偵測器時，您可以指定它的最大異常可見性期間。這是異常顯示在主控台中，並由 ListAnomalies API 操作傳回的天數。經過這段時間之後，如果持續發生異常，它會自動接受為一般行為，且異常偵測器模型會停止將其標記為異常。

如果您在建立異常偵測器時未調整可見性時間，則會使用 21 天做為預設值。

隱藏異常

發現異常之後，您可以選擇暫時或永久隱藏該異常。隱藏異常會導致異常偵測器在您指定的時間內停止將此出現標記為異常。當您抑制異常時，您可以選擇僅隱藏該特定異常，或抑制與發現異常的模式相關的所有異常。

您仍然可以在 主控台中檢視隱藏的異常。您也可以選擇停止隱藏它們。

常見問答集

是否 AWS 使用我的資料來訓練機器學習演算法，以供其他客戶 AWS 使用？

否。訓練建立的異常偵測模型是根據日誌群組中的日誌事件，並且僅在該日誌群組和該 AWS 帳戶中使用。

哪些類型的日誌事件適用於異常偵測？

日誌異常偵測非常適合：應用程式日誌和其他類型的日誌，其中大多數日誌項目都符合典型模式。包含日誌層級或嚴重性關鍵字的日誌群組，例如 INFO、R ERROR 和 DEBUG，特別適合用於日誌異常偵測。

日誌異常偵測不適用於：具有極長 JSON 結構的日誌事件，例如 CloudTrail Logs。模式分析最多只會分析日誌行的前 1500 個字元，因此超過該限制的任何字元都會略過。

稽核或存取日誌，例如 VPC 流程日誌，也會減少異常偵測的成功率。異常偵測旨在尋找應用程式問題，因此可能不適合網路或存取異常。

為了協助您判斷異常偵測器是否適合特定日誌群組，請使用 CloudWatch Logs 模式分析來尋找群組中日誌事件中的模式數目。如果模式數量不超過約 300，異常偵測可能運作良好。如需模式分析的詳細資訊，請參閱 模式分析。

哪些項目被標記為異常？

下列情況可能會導致日誌事件標記為異常：

- 具有先前在日誌群組中看不到模式的日誌事件。
- 已知模式的顯著變化。
- 動態字符的新值，具有一組分散的一般值。
- 動態字符值的出現次數發生大幅變更。

雖然上述所有項目都可能標記為異常，但它們並不表示應用程式效能不佳。例如，higher-than-usual200的成功值數目可能會標記為異常。在這種情況下，您可能會考慮抑制這些不代表問題的異常。

正在遮罩的敏感資料會發生什麼情況？

任何被遮罩為敏感資料的日誌事件部分都不會掃描異常。如需遮罩敏感資料的詳細資訊，請參閱使用遮罩協助保護敏感日誌資料。

在 CloudWatch Logs Insights 中使用異常偵測

除了建立日誌異常偵測器以進行持續監控之外，您也可以在 CloudWatch Logs Insights 查詢中使用anomaly命令，以隨需識別日誌資料中的異常模式。此命令擴展現有的pattern功能，並使用機器學習來偵測五種異常類型，包括模式頻率變更、新模式和字符變化。

anomaly 命令特別適用於：

- 臨機操作分析歷史日誌資料，以識別異常模式
- 調查異常行為的特定時段
- 監控 Lambda 函數等應用程式是否有執行問題

如需在查詢中使用 anomaly命令的詳細資訊，請參閱 異常。

此查詢型異常偵測可補充以下各節所述的連續異常偵測器，為您提供即時監控和隨需分析功能。

在日誌群組上啟用異常偵測

使用下列步驟來使用 CloudWatch 主控台建立日誌異常偵測器，以掃描日誌群組是否有異常。

您也可以以程式設計方式建立異常偵測器。如需詳細資訊，請參閱 CreateLogAnomalyDetector。

建立日誌異常偵測器

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 選擇日誌、日誌異常。
3. 選擇建立異常偵測器。
4. 選取要為其建立此異常偵測器的日誌群組。
5. 在異常偵測器名稱中輸入偵測器的名稱。
6. （選用）將評估頻率從預設值變更為 5 分鐘。根據日誌群組接收新日誌的頻率來設定此值。例如，如果日誌群組每 10 分鐘批次接收新的日誌事件，則將評估頻率設定為 15 分鐘可能是適當的。
7. （選用）若要將異常偵測器設定為僅在包含特定單字或字串的日誌事件中尋找異常，請選擇篩選模式。

   然後，以異常偵測篩選條件模式輸入模式。如需模式語法的詳細資訊，請參閱用於指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法。

   （選用）若要測試篩選條件模式，請在日誌事件訊息中輸入一些日誌訊息，然後選擇測試模式。
8. （選用）若要從預設變更異常可見性期間，或將 AWS KMS 金鑰與此異常偵測器建立關聯，請選擇進階組態。

   a. 若要從預設值變更異常可見性期間，請在異常可見性期間上限（天）中輸入新值。

b. 若要將 AWS KMS 金鑰與此異常偵測器建立關聯，請在 KMS 金鑰 ARN 中輸入 ARN。如果您指派金鑰，此偵測器找到的異常資訊會使用金鑰進行靜態加密。使用者必須擁有此金鑰和異常偵測器的許可，才能擷取找到的異常相關資訊。

   您還必須確保 CloudWatch Logs 服務主體具有使用金鑰的許可。如需詳細資訊，請參閱使用加密異常偵測器及其結果 AWS KMS。

9. 選擇啟用異常偵測。

   系統會建立異常偵測器，並根據日誌群組正在擷取的日誌事件開始訓練其模型。大約 15 分鐘後，異常偵測會處於作用中狀態，並開始尋找和顯示異常。

檢視找到的異常

建立一或多個日誌異常偵測器之後，您可以使用 CloudWatch 主控台來檢視他們找到的異常。

您可以以程式設計方式檢視異常。如需詳細資訊，請參閱 ListAnomalies。

檢視所有日誌異常偵測器找到的異常

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 選擇日誌、日誌異常。

   Logs 異常資料表隨即出現。日誌異常旁邊的頂端數字會顯示資料表中列出多少日誌異常。表格中的每一列都會顯示下列資訊：

   - 異常資料欄會顯示異常的簡短摘要。這些摘要是由 CloudWatch Logs 產生。
   - 異常的優先順序。根據日誌事件中的變更量、日誌事件中`Exception`發生的關鍵字等，自動計算優先順序。
   - 異常依據的日誌模式。如需模式的詳細資訊，請參閱 日誌異常偵測。
   - 異常日誌趨勢會顯示長條圖，描述符合模式的日誌量。
   - 上次偵測時間會顯示最近發現此異常的時間。
   - 第一次偵測時間會顯示第一次發現此異常。
   - 異常偵測器會顯示日誌群組的名稱，其中包含與此異常相關的日誌事件。您可以選擇此名稱以查看日誌群組詳細資訊頁面。
3. 若要進一步檢查一個異常，請選擇其列中的選項按鈕。

   模式檢查窗格隨即出現，並顯示下列項目：

- 此異常依據的模式。在模式中選取權杖，以分析該權杖的值。
- 顯示查詢時間範圍內異常出現次數的長條圖。
- 日誌範例索引標籤會顯示一些屬於異常的日誌事件。
- 如果您已選取動態權杖，權杖值索引標籤會顯示所選動態權杖的值。

> **Note**
>
> 每個字符最多擷取 10 個字符值。字符計數可能不精確。CloudWatch Logs 使用機率計數器來產生字符計數，而不是絕對值。

4. 若要隱藏異常，請選擇其列中的選項按鈕，然後執行下列動作：

   a. 選擇動作、隱藏異常。

   b. 然後指定您希望異常被抑制的時間長度。

   c. 若要隱藏與此模式相關的所有異常，請選取隱藏模式。

   d. 選擇隱藏異常。

檢視單一日誌群組中找到的異常

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 選擇日誌、日誌群組。
3. 選擇日誌群組的名稱，然後選擇異常偵測索引標籤。

   異常偵測資料表隨即出現。日誌異常旁邊的頂端數字會顯示資料表中列出多少日誌異常。表格中的每一列都會顯示下列資訊：

   - 異常資料欄會顯示異常的簡短摘要。這些摘要是由 CloudWatch Logs 產生。
   - 異常的優先順序。根據日誌事件中的變更量、日誌事件中`Exception`發生的關鍵字等，自動計算優先順序。
   - 異常依據的日誌模式。如需模式的詳細資訊，請參閱 日誌異常偵測。
   - 異常日誌趨勢會顯示長條圖，描述符合模式的日誌量。
   - 上次偵測時間會顯示最近發現此異常的時間。
   - 第一次偵測時間會顯示第一次發現此異常。

4. 若要進一步檢查一個異常，請選擇其列中的選項按鈕。

模式檢查窗格隨即出現，並顯示下列項目：

- 此異常依據的模式。在模式中選取權杖，以分析該權杖的值。
- 顯示查詢時間範圍內異常出現次數的長條圖。
- 日誌範例索引標籤會顯示一些屬於異常的日誌事件。
- 如果您已選取動態權杖，權杖值索引標籤會顯示所選動態權杖的值。

> **Note**
>
> 每個字符最多擷取 10 個字符值。字符計數可能不精確。CloudWatch Logs 使用機率計數器來產生字符計數，而不是絕對值。

5. 若要隱藏異常，請選擇其列中的選項按鈕，然後執行下列動作：
   a. 選擇動作、隱藏異常。
   b. 然後指定您希望異常被抑制的時間長度。
   c. 若要隱藏與此模式相關的所有異常，請選取隱藏模式。
   d. 選擇隱藏異常。

在日誌異常偵測器上建立警示

您可以在日誌群組中為日誌異常偵測器建立警示。您可以在指定的期間內，在日誌群組中找到指定數量的異常時，指定 讓警示進入 ALARM 狀態。您也可以使用篩選條件，讓警示只會計算指定優先順序的異常。

為日誌異常偵測器建立警示

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌、日誌異常。

   日誌異常偵測器的資料表隨即出現。
3. 選擇您要為其設定警示之異常偵測器的選項按鈕，然後選擇建立警示。

   CloudWatch 警示建立精靈隨即出現。LogAnomalyDetector 欄位會顯示您選擇的異常偵測器名稱。指標名稱欄位會顯示AnomalyCount。
4. （選用）若要針對異常優先順序篩選此警示，請執行下列其中一項操作：

- 若要讓警示只計數高優先順序異常，請輸入 LogAnomalyPriority **HIGH**的 。
- 若要只讓警示計數高優先順序和中優先順序異常，請輸入 LogAnomalyPriority **MEDIUM**的 。

如需優先順序層級的詳細資訊，請參閱 異常和模式的嚴重性和優先順序。

5. 選擇使用警示的靜態或指標異常偵測閾值。此選擇決定警示閾值的設定方式。靜態閾值表示警示閾值是您選擇的靜態常數。異常偵測閾值表示 CloudWatch 會決定一般值的範圍，如果實際計數超過此頻帶的閾值，則警示會觸發。您不需要為日誌異常偵測警示選擇異常偵測。如需指標異常偵測的詳細資訊，請參閱使用 CloudWatch 異常偵測。
6. 對於當 ***your-metric-name*** 為...，選擇大於、大於/等於、小於/等於或小於。對於相比...，為閾值指定一個數字。如果異常偵測器在期間指定的時間內發現超過此數量的警示，則警示會進入 ALARM 狀態。
7. 選擇 Additional configuration (其他組態)。針對 Datapoints to alarm (要警示的資料點)，請指定 (資料點) 必須處於 ALARM 狀態多少評估期間，才會觸發警示。如果此處的兩個值相符，您便可以建立警示，在許多連續期間違規時移至 ALARM 狀態。

   若要建立 N 個中有 M 個警示，請針對小於第二個值之數字的第一個值指定數字。如需詳細資訊，請參閱評估警示。
8. 對於 Missing data treatment (遺失資料處理方式)，選擇警示在遺失某些資料點時的行為。如需詳細資訊，請參閱設定 CloudWatch 警示如何處理遺失的資料。
9. 選擇下一步。
10. 針對通知，選擇新增通知，然後指定警示轉換為 ALARM、 OK或 INSUFFICIENT_DATA 狀態時要通知的 Amazon SNS 主題。

    a. (選用) 若要針對相同警示狀態或不同警示狀態傳送多個通知，請選擇 Add notification (新增通知)。

    > **Note**
    >
    > 建議您設定警示，以便除了在進入警示狀態外，進入資料不足狀態時應採取動作。這是因為連線至資料來源的 Lambda 函數有許多問題可能會導致警示轉換為資料不足。

    b. (選用) 若不傳送 Amazon SNS 通知，請選擇移除。
11. （選用） 如果您希望警示針對 Amazon EC2 Auto Scaling、Amazon EC2、票證或 執行動作 AWS Systems Manager，請選擇適當的按鈕，並指定警示狀態和動作。

> **Note**
>
> 警示僅在處於 ALARM 狀態時執行 Systems Manager 動作。如需有關 Systems Manager 動作的資訊，請參閱設定 CloudWatch 以建立 OpsItems 和事件建立。

12. 選擇下一步。
13. 在 Name and description (名稱和描述) 下，輸入警示的名稱和描述，然後選擇 Next (下一步)。此名稱只能包含 UTF-8 字元，不能包含 ASCII 控制字元。說明可以包括 Markdown 格式，僅在 CloudWatch 主控台的警示詳細資訊標籤中顯示。Markdown 對於將連結新增至執行手冊或其他內部資源很實用。

> **Tip**
>
> 警示名稱只能包含 UTF-8 字元。它不能包含 ASCII 控制字元。

14. 在 Preview and create (預覽及建立) 下，請確認警示資訊和條件都是正確的，然後選擇 Create alarm (建立警示)。

日誌異常偵測器發佈的指標

CloudWatch Logs 會將 AnomalyCount 指標發佈至 CloudWatch 指標。此指標會發佈至 AWS/Logs 命名空間。

AnomalyCount 指標會以下列維度發佈：

- LogAnomalyDetector – 異常偵測器的名稱
- LogAnomalyPriority – 異常的優先順序層級

使用 加密異常偵測器及其結果 AWS KMS

異常偵測器資料一律會在 CloudWatch Logs 中加密。根據預設，CloudWatch Logs 會對靜態資料使用伺服器端加密。或者，您可以使用 AWS Key Management Service 進行此加密。如果您這麼做，則會使用 AWS KMS 金鑰完成加密。透過將 KMS 金鑰與異常偵測器建立關聯，在異常偵測器層級啟用使用 AWS KMS 加密。

> **Important**
>
> CloudWatch Logs 僅支援對稱 KMS 金鑰。請勿使用非對稱金鑰來加密日誌群組中的資料。如需詳細資訊，請參閱使用對稱和非對稱金鑰。

限制

- 若要執行下列步驟，您必須擁有下列許可：kms:CreateKey、kms:GetKeyPolicy 和 kms:PutKeyPolicy。
- 將金鑰與異常偵測器建立關聯或取消關聯後，操作最多可能需要五分鐘才會生效。
- 如果您撤銷 CloudWatch Logs 對已關聯金鑰的存取權，或刪除所關聯的 KMS 金鑰，您將無法再擷取 CloudWatch Logs 中的加密資料。

步驟 1：建立 AWS KMS 金鑰

若要建立 KMS 金鑰，請使用以下 create-key 命令：

```
aws kms create-key
```

輸出包含金鑰 ID 和金鑰的 Amazon Resource Name (ARN)。下列為範例輸出：

```
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "key-default-1",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1478910250.94,
        "Arn": "arn:aws:kms:us-west-2:123456789012:key/key-default-1",
        "AWSAccountId": "123456789012",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
```

```
}
```

步驟 2：設定 KMS 金鑰許可

根據預設，所有 AWS KMS 金鑰都是私有的。只有資源擁有者可以使用它來加密和解密資料。然而，資源擁有者可以授與其他使用者和資源存取 KMS 金鑰的許可。在此步驟中，您會給予 CloudWatch Logs 服務主體使用金鑰的許可。此服務主體必須位於存放 KMS 金鑰的相同 AWS 區域中。

根據最佳實務，建議您將 KMS 金鑰的使用限制為您指定的 AWS 帳戶或異常偵測器。

首先，使用以下 get-key-policy 命令，將 KMS 金鑰的預設政策儲存為 `policy.json`：

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

在文字編輯器中開啟 `policy.json` 檔案，並從下列其中一個陳述式中加入區段 (以粗體顯示)。使用逗號從新陳述式中分隔現有陳述式。這些陳述式使用`Condition`區段來增強 AWS KMS 金鑰的安全性。如需詳細資訊，請參閱AWS KMS 金鑰和加密內容。

此範例中的 `Condition`區段會將金鑰 AWS KMS 的使用限制在指定的帳戶，但可用於任何異常偵測器。

JSON

```
{
"Version":"2012-10-17",
"Id": "key-default-1",
"Statement": [
    {
        "Sid": "EnableIAMUserPermissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:root"
        },
        "Action": "kms:*",
        "Resource": "*"
    },
    {
        "Sid": "AllowCloudWatchLogsEncryption",
        "Effect": "Allow",
        "Principal": {
```

```
                "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:GenerateDataKey*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-
east-1:123456789012:anomaly-detector:*"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:logs:us-
east-1:123456789012:anomaly-detector:*"
                }
            }
        },
        {
            "Sid": "AllowCloudWatchLogsDescribeKey",
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": "kms:DescribeKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        },
        {
            "Sid": "AllowCloudWatchLogsReEncryption",
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
```

```
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws-crypto-ec:aws:logs:arn":
 "arn:aws:logs:us-east-1:123456789012:anomaly-detector:*"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:logs:us-
east-1:123456789012:anomaly-detector:*"
                }
            }
        },
        {
            "Sid": "AllowCloudWatchLogsDescribeKeyForReEncryption",
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": "kms:DescribeKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

最後，使用下列 put-key-policy 命令新增更新的政策：

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

步驟 3：將 KMS 金鑰與異常偵測器建立關聯

您可以在主控台或使用 或 AWS CLI APIs 建立 KMS 金鑰時，將 KMS 金鑰與異常偵測器建立關聯。

步驟 4：取消金鑰與異常偵測器的關聯

金鑰與異常偵測器建立關聯後，您就無法更新金鑰。移除金鑰的唯一方法是刪除異常偵測器，然後重新建立它。

使用 CloudWatch Logs Live Tail 進行故障診斷

CloudWatch Logs Live Tail 可在擷取時檢視新日誌事件的串流清單，協助您快速進行事件疑難排解。可以近乎即時地檢視、篩選和反白顯示擷取的日誌，協助您快速偵測並解決問題。可以根據指定的詞彙篩選日誌，並反白顯示包含指定詞彙的日誌，以協助您快速找到查詢內容。

Live Tail 工作階段會按工作階段使用時間 (每分鐘) 產生費用。如需定價的詳細資訊，請參閱 Amazon CloudWatch 定價中的日誌索引標籤。

僅標準日誌類別中的日誌群組支援 Live Tail。如需日誌類別的詳細資訊，請參閱 日誌類別。

下列各節說明如何在 主控台和 中使用 Live Tail AWS CLI。您也可以以程式設計方式啟動 Live Tail 工作階段。如需詳細資訊，請參閱 StartLiveTail。如需 SDK 範例，請參閱使用 AWS SDK 啟動 Live Tail 工作階段。

您也可以在 中使用 Live Tail AWS Toolkit for Visual Studio Code。若要從 VS Code Command Palette 啟動 Live Tail 工作階段，請參閱 AWS Toolkit for Visual Studio Code 《 使用者指南》中的 Amazon CloudWatch Logs Live Tail 一節。

Live Tail 功能適用於所有 commercial AWS Regions。它不適用於中國區域或 AWS GovCloud (US) 區域。

> **Note**
>
> `StartLiveTail` API 會使用 SDK 主機字首注入來路由請求。2026 年 4 月 1 日之前發行的 SDK 版本會路由至 `streaming-logs.`*`Region`*`.amazonaws.com`，這不支援 VPC 端點。2026 年 4 月 1 日當天或之後發行的 SDK 版本會路由至 `stream-logs.`*`Region`*`.amazonaws.com`，以支援 VPC 端點。若要為此 API 設定 VPC 端點，請參閱為 CloudWatch Logs 建立 VPC 端點。

使用 啟動 Live Tail 工作階段 AWS CLI

`start-live-tail` AWS CLI 命令會啟動終端機中一或多個日誌群組的 Live Tail 串流工作階段。Live Tail 工作階段最多可持續三個小時。如果每秒超過 500 個日誌事件符合篩選條件，則顯示的日誌事件是總日誌事件的範例，以提供即時結尾體驗。如需 `start-live-tail`命令的詳細資訊，請參閱 start-live-tail

您可以在start-live-tail兩種模式中使用 ：

- 僅限列印 – 這是預設模式
- 互動式

僅列印

在 print-only模式下，日誌事件會在終端機上串流。每秒會在底部新增新事件，建立與 Linux tail -f上類似的近乎即時結尾體驗。

若要以僅列印模式啟動 Live Tail 工作階段，請輸入下列命令。

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs
```

當您使用僅限列印模式時，也可以將其與其他 Linux 命令進行管道傳輸，以提高其分析功能。下列範例會使用 error關鍵字篩選日誌事件，並列印這些事件的第二欄和第四欄，以協助您擷取特定資訊。

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs --mode print-only | grep "error" | awk '{print
 $2, $4}'
```

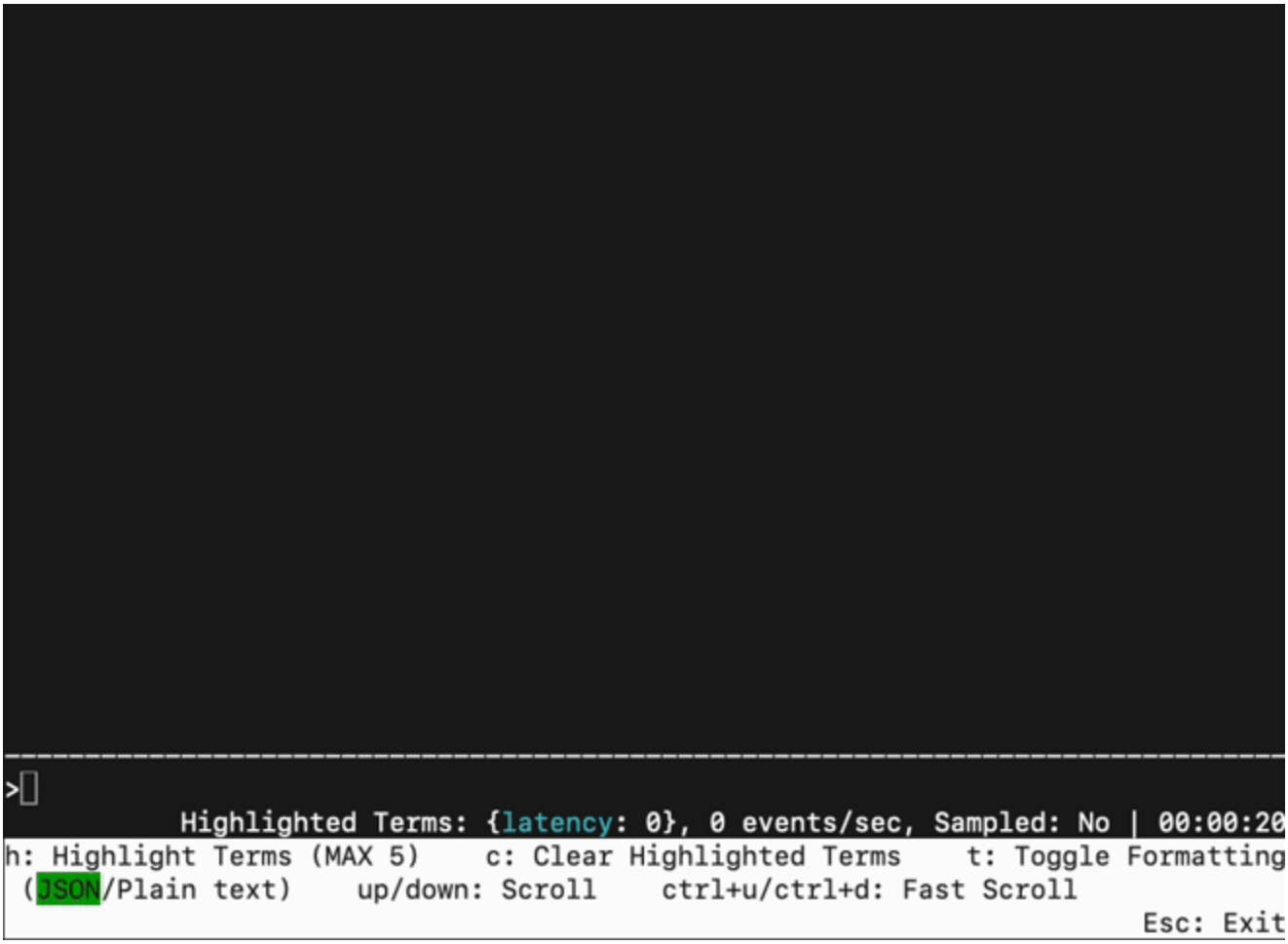
互動

在 interactive模式中，您可以反白詞彙，並在 JSON 和純文字之間切換輸出日誌事件的格式。互動式模式也會顯示 Live Tail 工作階段的相關資訊，例如工作階段持續時間、工作階段是否正在取樣，以及目前反白顯示的詞彙，以及遇到的次數。

若要以互動式模式啟動 Live Tail 工作階段，請輸入下列命令。

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs --mode interactive
```

Live Tail 工作階段開始。下列影片顯示範例工作階段的一部分。



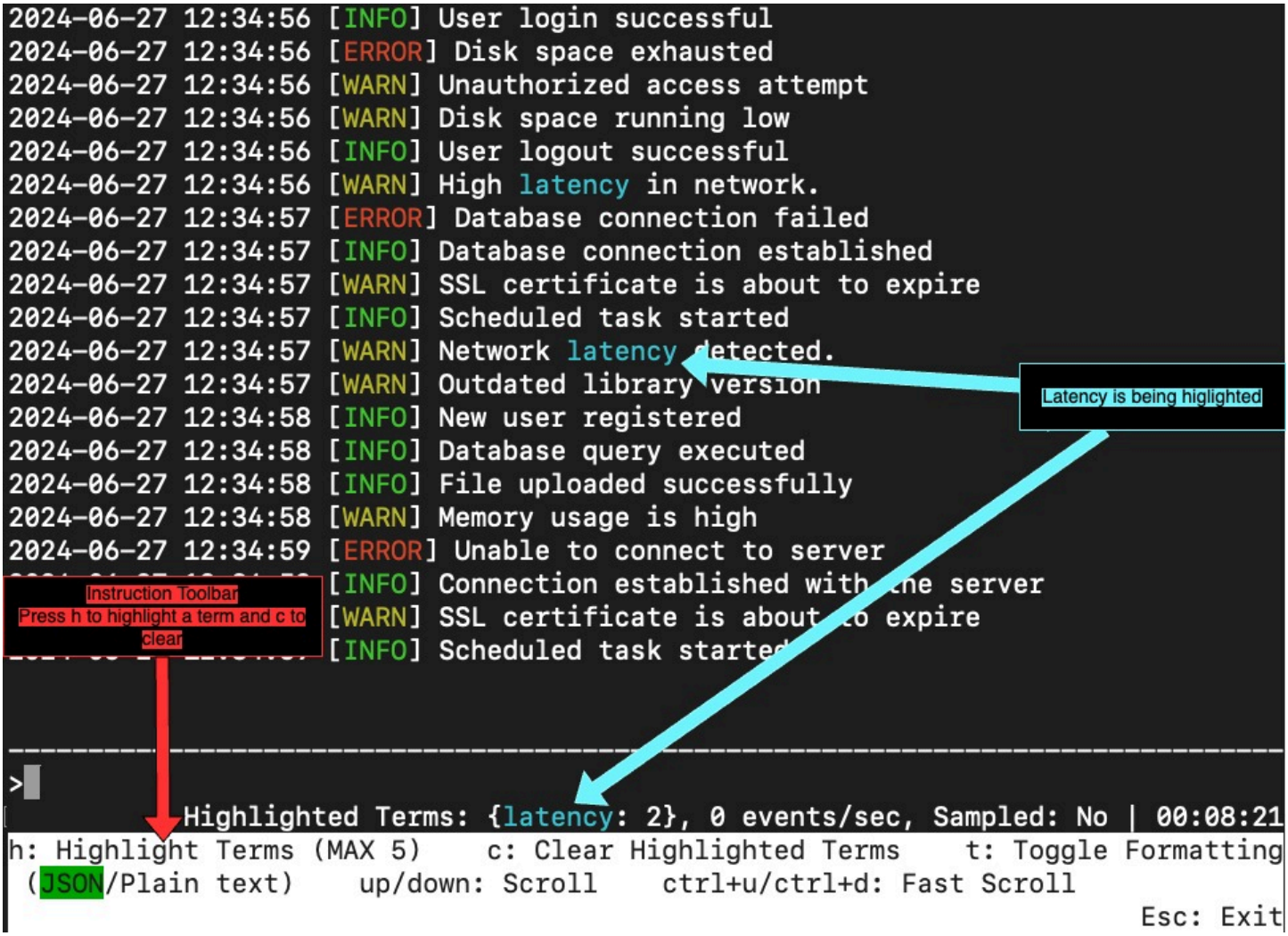
若要在串流日誌中反白詞彙，請按 h，然後輸入詞彙。以下顯示latency已反白顯示詞彙後的畫面。

若要清除反白詞彙，請按 c，然後輸入代表您要停止反白詞彙的數字。

您可以按 t 在 JSON 和純文字之間切換傳入事件的顯示格式。此切換功能會盡最大努力，只有在日誌事件格式相容時才會發生。

您可以使用向上和向下箭頭鍵來捲動，並使用 CTRL+u和 CTRL+d 來更快速地捲動。

下圖顯示 Live Tail 工作階段期間latency詞彙的反白。



在主控台中啟動 Live Tail 工作階段

您可以使用 CloudWatch 主控台來啟動 Live Tail 工作階段。下列程序說明如何使用左側導覽窗格中的 Live tail 選項來啟動 Live Tail 工作階段。您也可以從「日誌群組」頁面或 CloudWatch Logs Insights 頁面啟動 Live Tail 工作階段。

如果您使用資料保護政策來遮罩透過 Live Tail 檢視之日誌群組中的敏感資料，則敏感資料在 Live Tail 工作階段中顯示為遮罩狀態。如需有關遮罩日誌群組敏感資料的詳細資訊，請參閱 使用遮罩功能協助保護敏感日誌資料。

> **Important**
>
> 如果您的網路安全團隊不允許使用 Web 通訊端，您目前無法存取 CloudWatch 主控台的 Live Tail 部分。您可以使用 Live Tail 搭配 AWS CLI 或 APIs。如需詳細資訊，請參閱 使用 啟動 Live Tail 工作階段 AWS CLI和 StartLiveTail。

開始 Live Tail 工作階段

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌，然後選擇 Live tail。
3. 針對選取日誌群組，在 Live Tail 工作階段中，選取您要從中檢視事件的日誌群組。您最多可以選取 10 個日誌群組。
4. (選用) 如果您只選取一個日誌群組，則可以選取一個或多個日誌串流來檢視日誌事件，從而進一步篩選 Live Tail 工作階段。若要這樣做，請在選取日誌串流下，從下拉式清單中選取日誌串流的名稱。或者，您也可以使用選取日誌串流下的第二個方塊輸入日誌串流名稱字首，然後選取名稱與該字首相符的所有日誌串流。
5. (選用) 若要僅顯示包含特定字詞或其他字串的日誌事件，請在 `Add filter patterns` 中輸入字詞或字串。

   例如，若僅顯示包含 `Warning` 字詞的日誌事件，請輸入 **`Warning`**。篩選條件欄位區分大小寫。可以在此欄位中包含多個詞彙和模式運算子。

   - **`error 404`** 僅顯示包含 `error` 和 `404` 的日誌事件
   - **`?Error ?error`** 顯示包含 `Error` 或 `error` 的日誌事件
   - **`-INFO`** 顯示不包含 `INFO` 的所有日誌事件
   - **`{ $.eventType = "UpdateTrail" }`** 顯示事件類型欄位值為 `UpdateTrail` 的所有 JSON 日誌事件

   您也可以使用規則表達式 (regex) 來篩選：

   - **`%ERROR%`** 使用 regex 顯示包含 ERROR 關鍵字的所有日誌事件
   - **`{ $.names = %Steve% }`** 使用 regex 顯示 Steve 為 "name" 屬性中的 JSON 日誌事件
   - **`[ w1 = %abc%, w2 ]`** 使用 regex 顯示以空格分隔且第一個單詞為 abc 的日誌事件

如需有關模式語法的詳細資訊，請參閱篩選條件模式語法。

6. (選用) 若要反白顯示某些顯示的日誌事件，請輸入要搜尋的詞彙，並在 Live Tail 下反白顯示。一次輸入一個反白顯示詞彙。如果新增多個詞彙進行反白顯示，則會指派不同的顏色來代表每個詞彙。反白顯示指示器會顯示在任何包含指定詞彙的日誌事件的左側，當您展開主視窗中的日誌事件以檢視完整日誌事件時，也會出現在詞彙本身下方。

   您可以使用篩選功能以及反白顯示來快速疑難排解問題。例如，您可以篩選事件以僅顯示包含 `Error` 的事件，然後反白顯示包含 `404` 的事件。

7. 若要啟動工作階段，請選擇套用篩選條件

   相符的日誌事件開始出現在視窗中。也會顯示下列資訊：

   - 計時器會顯示 Live Tail 工作階段已啟用的時間長度。
   - 事件數/秒會顯示每秒有多少個擷取的日誌事件與您設定的篩選條件相符。
   - 由於許多事件與篩選條件相符，為了避免工作階段捲動速度過快，CloudWatch Logs 可能只會顯示一些相符的事件。如果發生這種情況，螢幕上顯示的相符事件百分比會以 % 的形式顯示。

8. 若要暫停事件流程以調查目前顯示的內容，請按一下事件視窗中的任意位置。
9. 在工作階段期間，您可以使用下列項目來查看有關每個日誌事件的詳細資訊。

   - 若要在主視窗中顯示日誌事件的完整文字，請選擇該日誌事件旁邊的箭頭。
   - 若要在側視窗中顯示日誌事件的完整文字，請選擇該日誌事件旁邊的 + 放大鏡。事件流程會暫停，並顯示側視窗。

     在側視窗中顯示日誌事件文字非常有用，可比較其文字與主視窗中的其他事件。

10. 若要停止 Live Tail 工作階段，請選擇停止。
11. 若要重新啟動工作階段，可選擇使用篩選面板修改篩選條件，然後選擇套用篩選條件。然後選擇 Start (啟動)。

跨帳戶跨區域日誌集中化

Amazon CloudWatch Logs 資料集中使用 AWS Organizations ，使用跨帳戶和跨區域集中化規則，從多個成員帳戶收集日誌資料到單一資料儲存庫。可以定義規則，將日誌資料從多個帳戶和 AWS 區域自動複寫到組織內的集中帳戶。此功能可簡化日誌整合，以改善整個 AWS 基礎設施的集中式監控、分析和合規。

CloudWatch Logs 資料集中化提供組態彈性，以滿足操作和安全性需求，例如在目的地帳戶內的規則設定期間設定備份區域的能力，以確保更高的彈性。此外，您可以完全控制從來源帳戶複製的日誌群組的加密行為，以處理最初使用客戶受管 KMS 金鑰加密的資料。

> **Note**
>
> CloudWatch Logs 集中化功能只會在您建立集中化規則後，處理抵達來源帳戶的新日誌資料。歷史日誌資料 （ 規則建立之前已存在的日誌 ） 不會集中。

資料集中化概念

開始使用 CloudWatch Logs 資料集中化之前，請先熟悉下列概念：

集中化規則

定義如何將來源帳戶和區域的日誌資料複寫至目的地帳戶和區域的組態。規則指定來源條件和目的地設定。

來源帳戶

日誌資料產生的 AWS 帳戶。來源帳戶的日誌事件會根據您定義的集中化規則複寫到目的地帳戶。

目的地帳戶

儲存複寫日誌資料的目的地 AWS 帳戶。此帳戶可做為日誌分析和監控的集中位置。

備份區域

目的地帳戶中的選用次要區域，可複寫日誌資料，以提高彈性和災難復原目的。

CloudWatch Logs 中的加密

在 CloudWatch Logs 中，日誌群組資料一律會加密。根據預設，CloudWatch Logs 會使用伺服器端加密搭配 256 位元進階加密標準 Galois/計數器模式 (AES-GCM) 來加密靜態日誌資料。或者，您

可以使用 AWS Key Management Service 進行此加密。如需詳細資訊，請參閱 CloudWatch Logs 加密文件。

- 集中化期間加密的運作方式：CloudWatch Logs 集中化會在擷取時間主動將日誌資料從來源帳戶複製到目的地帳戶。在此過程中，您的資料會使用 AWS 擁有的服務金鑰在傳輸中保持加密。來源和目的地日誌群組中的靜態資料會使用您選擇的加密方法（客戶管理或 AWS 擁有的 KMS 金鑰）進行加密。如果您在目的地日誌群組中使用客戶受管 KMS 金鑰，請將標籤新增至 `LogsManaged = true` kms 金鑰，供集中化服務存取。
- 需要 KMS 許可時：
  - 如果您在來源帳戶中使用客戶受管 KMS 金鑰，CloudWatch Logs 在下列範例案例中需要 KMS 許可：
    - 輸送量管理：達到集中式輸送量限制時，系統會使用客戶受管 KMS 金鑰暫時加密日誌資料，直到頻寬可用為止。
    - 資料保護和修訂：當來源日誌群組啟用資料保護政策時，CloudWatch Logs 需要解密許可才能存取原始日誌資料以集中。

> **Important**
>
> 集中化規則由 AWS Organizations 管理帳戶或委派管理員管理。若要從集中化中排除客戶受管 KMS 加密的日誌群組，請將規則設定設為「不要集中使用 AWS KMS 金鑰加密的日誌群組」。

設定日誌集中

若要設定 CloudWatch Logs Centralization，您需要設定集中化規則，以定義日誌資料如何從來源帳戶中的日誌群組流向目的地帳戶中的日誌群組。

啟用集中化規則並將日誌事件複寫至目的地帳戶後，您就可以在具有增強篩選功能的集中式日誌群組上建立指標、訂閱和帳戶篩選條件。這些篩選條件可鎖定來自特定來源帳戶和區域的日誌事件，並可發出來源帳戶和區域資訊做為指標維度。如需詳細資訊，請參閱使用篩選條件從日誌事件建立指標。

先決條件

- AWS Organizations 必須設定 ，且來源和目的地帳戶必須都屬於組織。
- 必須為 CloudWatch、管理帳戶和目的地帳戶啟用信任存取，以便提供對日誌資料的存取。

> **Note**
>
> 建議您透過主控台啟用受信任存取，以自動建立所需的服務連結角色 (SLR)。如果透過其他方法啟用受信任存取，則需要單獨建立服務連結角色。

自訂目的地日誌群組名稱

建立集中化規則時，您可以使用 屬性自訂目的地日誌群組名稱的結構方式。建立日誌群組時，這些屬性會自動取代為實際值，讓您可以在目的地帳戶中以階層方式組織日誌。根據預設，只會使用 `${source.logGroup}` 屬性，該屬性會合併目的地帳戶中名稱相同的所有日誌群組。如果變數無法解析，它會繼承階層中其父變數的值。

可用的屬性

您可以在目的地日誌群組名稱模式中使用下列屬性：

目的地日誌群組名稱屬性

屬性	Description
`${source.accountId}`	日誌來源 AWS 的帳戶 ID。
`${source.region}`	AWS 區域 日誌來源的 。
`${source.logGroup}`	來源帳戶的原始日誌群組名稱。
`${source.org.id}`	來源帳戶的 AWS Organizations ID。
`${source.org.ouId}`	來源帳戶的組織單位 ID
`${source.org.rootId}`	組織根 ID
`${source.org.path}`	從帳戶到根目錄的完整組織路徑

範例

保留原始日誌群組結構

模式：`/centralized/${source.accountId}${source.logGroup}`

結果：/centralized/123456789012/aws/lambda/my-function

依帳戶和區域組織

模式：/centralized/${source.accountId}/${source.region}

結果：/centralized/123456789012/us-east-1

依組織結構組織

模式：/logs/${source.org.id}/${source.org.ouId}/${source.accountId}

結果：/logs/o-abc123/ou-xyz-12345678/123456789012

簡單平面結構

模式：/centralized-logs

結果：/centralized-logs

最佳實務

- 包含來源帳戶 ID，以輕鬆識別來自哪些帳戶日誌。
- 如果您要從多個區域集中化，請包含來源區域。
- 結構目的地日誌群組名稱少於 512 個字元。CloudWatch Logs 強制執行最多 512 個字元的日誌群組名稱長度。

建立集中化規則

使用下列程序建立集中化規則，將日誌資料從來源帳戶複寫到您的目的地帳戶。

建立集中化規則

1. 導覽至組織管理或委派管理員帳戶中的 CloudWatch 主控台。
2. 選擇設定。
3. 導覽至組織索引標籤。
4. 選擇設定規則。
5. 設定下列欄位來指定來源詳細資訊，然後選擇下一步：

    a. 集中化規則名稱：輸入集中化規則的唯一名稱。

b. 來源帳戶：定義來源選擇條件，以挑選將集中遙測資料的帳戶。選擇條件可能包括：

- 組織中的成員帳戶清單
- 組織中組織單位的清單
- 整個組織

您可以透過兩種模式提供選擇條件：

- 建置器：產生來源選取條件的點擊式體驗
- 編輯器：自由格式文字方塊，提供來源選取條件

來源選取條件支援的語法：

- 支援的金鑰：OrganizationId | OrganizationUnitId | AccountId | *
- 支援的運算子： = | IN | OR

c. 來源區域：選取要尋找要集中的遙測資料的區域清單。

6. 設定下列欄位來指定目的地詳細資訊，然後選擇下一步：

a. 目的地帳戶：選取組織中做為遙測資料中央目的地的帳戶。

b. 目的地區域：選取存放集中式遙測資料副本的主要區域。

c. 備份區域：選擇性地選取存放集中式遙測資料的第二個副本的區域。

7. 設定下列欄位來指定遙測資料，然後選擇下一步：

a. 日誌群組：選擇下列其中一個選項：

- 所有日誌群組：集中來源帳戶中所有日誌群組的日誌。
- 篩選日誌群組：將來源帳戶中日誌群組子集的日誌集中，符合選取條件。您可以透過兩種模式提供選擇條件：
  - 建置器：以點選為基礎的體驗，以產生選擇條件
  - 編輯器：提供選擇條件的自由格式文字方塊

  您可以使用兩種選擇條件來篩選日誌：

  - 日誌群組選取條件：指定要集中哪些來源日誌群組的選取條件。
    - 支援的金鑰：LogGroupName | *
    - 支援的運算子： = | ! = | IN | NOT IN | AND | OR | LIKE | NOT LIKE

- 資料來源選取條件：指定要集中哪些資料來源的選取條件。
  - 支援的金鑰：DataSourceName | DataSourceType
  - 支援的運算子： = | ！= | IN | NOT IN | AND | OR | LIKE | NOT LIKE

當同時指定日誌群組選取條件和資料來源選取條件時，日誌事件必須符合這兩個條件才能集中。

b. KMS 加密日誌群組

> **Important**
>
> 如果集中化規則中提供的 KMS 金鑰不允許 CloudWatch Logs 使用日誌，CloudWatch 集中化規則將無法將日誌從來源帳戶交付到目的地日誌群組。 CloudWatch 如果您在目的地日誌群組中使用客戶受管 KMS 金鑰，請將 LogsManaged = true 標籤新增至 kms 金鑰。如需詳細資訊，請參閱步驟 2：設定 KMS 金鑰許可。

請選擇下列其中一個選項：

- 使用目的地特定的客戶受管 KMS 金鑰，將以客戶受管 KMS 金鑰加密的來源日誌群組集中：將來自以客戶受管 KMS 金鑰加密的來源日誌群組的日誌事件集中到目的地帳戶中以客戶受管 KMS 金鑰加密的目的地日誌群組。

  選取此設定時，您還必須設定下列項目：

  - 目的地加密金鑰 ARN：目的地帳戶和主要目的地區域中客戶受管 KMS 金鑰的 ARN，要與新建立的目的地日誌群組建立關聯。
  - 備份目的地加密金鑰 ARN （如果選取備份區域）：目的地帳戶和備份目的地區域中客戶受管 KMS 金鑰的 ARN，要與新建立的目的地日誌群組建立關聯。
  - 略過集中至未加密的目的地日誌群組 （選用）：如果日誌群組已存在，而沒有客戶受管 KMS 金鑰，CloudWatch 就無法更新其加密。選擇此選項，將日誌事件從以客戶受管 KMS 金鑰加密的來源日誌群組集中，略過到與客戶受管 KMS 金鑰無關的目的地日誌群組。
- 使用 AWS 擁有的 KMS 金鑰將以客戶受管 KMS 金鑰加密的日誌群組集中化：將來自以客戶受管 KMS 金鑰加密的來源日誌群組的日誌事件集中到使用 AWS 擁有的 KMS 金鑰加密的新建立目的地日誌群組。

- 請勿集中使用客戶受管 KMS 金鑰加密的日誌群組：從使用客戶受管 KMS 金鑰加密的來源日誌群組略過日誌事件的集中化。

8. 檢閱集中化規則，選擇性地進行任何最後一分鐘編輯，然後選擇建立集中化政策。

修改集中化規則

使用下列程序來修改現有的集中化規則。

修改集中化規則

1. 導覽至組織管理或委派管理員帳戶中的 CloudWatch 主控台。
2. 選擇設定。
3. 導覽至組織索引標籤。
4. 選擇管理規則。
5. 選取要更新的規則，然後選擇編輯。
6. 視需要更新規則組態，選擇下一步以繼續每個步驟。
7. 在步驟 4 檢閱和設定中，選擇更新集中政策。

檢視集中化規則

使用下列程序來檢視現有集中化規則的詳細資訊。

檢視集中化規則

1. 導覽至組織管理或委派管理員帳戶中的 CloudWatch 主控台。
2. 選擇設定。
3. 導覽至組織索引標籤。
4. 選擇管理規則。
5. 檢視所有現有集中化規則的清單，然後選擇特定規則名稱以檢視其詳細資訊。

刪除集中化規則

使用下列程序刪除現有的集中化規則。

刪除集中化規則

1. 導覽至組織管理或委派管理員帳戶中的 CloudWatch 主控台。
2. 選擇設定。
3. 導覽至組織索引標籤。
4. 選擇管理規則。
5. 選取要刪除的規則，然後選擇刪除。
6. 確認刪除，然後選擇 Delete (刪除)。

監控和疑難排解集中化規則

您可以使用 CloudWatch 指標、CloudWatch Logs 主控台和 AWS CloudTrail 日誌來監控集中化規則的狀態和效能。這可協助您確保日誌資料已成功複寫，並識別集中化組態的任何問題。

CloudWatch Logs 提供：

1. 每個集中化規則的規則運作狀態
   a. 選擇設定。
   b. 導覽至組織索引標籤。
   c. 選擇管理規則。
2. 使用 記錄 API 呼叫 AWS CloudTrail
3. CloudWatch 也會發佈用於集中化的指標，包括複寫的日誌事件、錯誤和限流。如需這些指標及其維度的詳細資訊，請參閱 集中化指標和維度。

集中化規則運作狀態

每個集中化規則都有運作狀態，指出其是否正常運作。您可以透過主控台或以程式設計方式使用 API 檢查規則運作狀態。

規則運作狀態包括：

- HEALTHY：規則正常運作，並依設定複寫日誌資料
- UNHEALTHY：規則遇到問題，可能無法正確複寫資料
- PROVISIONING：組織的集中化正在進行設定。

當規則標示為 UNHEALTHY 時， FailureReason 欄位會提供需要解決之特定問題的詳細資訊。

使用 監控集中化 API 呼叫 AWS CloudTrail

AWS CloudTrail 會記錄對集中服務發出的 API 呼叫，讓您追蹤組態變更，並針對屬於 之帳戶的問題進行疑難排解 AWS Organizations。

集中化的關鍵 CloudTrail 事件包括：

- CreateCentralizationRuleForOrganization：建立新的集中化規則時
- UpdateCentralizationRuleForOrganization：修改現有規則時
- DeleteCentralizationRuleForOrganization：刪除規則時
- GetCentralizationRuleForOrganization：擷取規則詳細資訊時
- ListCentralizationRulesForOrganization：列出規則時

您可以使用 CloudTrail 日誌來稽核集中化組態變更，並將其與效能問題或複寫失敗相關聯。

監控建議

為了確保集中化正常運作，我們建議您在提供給 CloudWatch 指標的金鑰集中化指標上設定 CloudWatch 警示。此主動監控可協助您及早偵測問題，並在整個組織中維持可靠的日誌集中化。

要監控的關鍵指標包括：

- IncomingCopiedBytes：監控成功複寫至目的地帳戶的日誌資料量。突然下降或缺少此指標可能表示集中化問題。
- CentralizationError：針對集中化程序中的任何錯誤設定警示，以快速識別和解決問題。
- CentralizationThrottled：監控可能影響日誌複寫效能的限流事件。

如需可用集中化指標及其維度的完整清單，請參閱 集中化指標和維度。

如果日誌未如預期集中，請檢閱下列可能阻止日誌集中化的常見案例。

歷史日誌資料

CloudWatch Logs 集中化功能只會在您建立集中化規則後，處理抵達來源帳戶的新日誌資料。歷史日誌資料（規則建立之前已存在的日誌）不會集中。

KMS 金鑰許可

如果集中化規則中提供的 KMS 金鑰不允許 CloudWatch Logs 使用日誌，集中化規則將無法將日誌從來源帳戶交付到目的地日誌群組。確保 KMS 金鑰政策將必要的許可授予 CloudWatch Logs。如需詳細資訊，請參閱步驟 2：設定 KMS 金鑰許可。

客戶受管 KMS 金鑰組態

如果您選取在規則建立期間不集中使用客戶受管 KMS 金鑰加密的日誌群組，則會略過使用客戶受管 KMS 金鑰加密的來源日誌群組中的日誌事件，而不會集中。

目的地加密不相符

如果目的地日誌群組已存在與集中規則指定的不同 KMS 加密組態，且衝突解決設定為 SKIP，則會捨棄記錄並發出`DestinationEncryptionMismatch`錯誤。例如，當目的地具有預設加密，但規則指定客戶受管 KMS 金鑰時，就會發生這種情況。

未啟用信任的存取

必須為 中的 CloudWatch 啟用受信任存取 AWS Organizations ，以便管理帳戶和目的地帳戶提供對日誌資料的存取。

來源選取條件

確認已正確設定集中化規則的來源選取條件：

- 帳戶和區域：確保日誌來源的來源帳戶和區域包含在規則中。來自規則中未指定之帳戶或區域的日誌群組將不會集中。
- 日誌群組篩選條件：如果您設定日誌群組篩選條件，則只會集中處理符合指定條件的日誌群組。確認您的日誌群組選擇條件包含您預期集中的日誌群組。
- 組織成員資格：來源和目的地帳戶都必須屬於同一個 AWS Organizations 組織。組織外部的帳戶無法參與集中化。

已達到日誌群組配額限制

如果目的地帳戶已達到其日誌群組配額限制，則無法建立新的日誌群組以進行集中。確認目的地帳戶有足夠的配額來容納來自所有來源帳戶的集中式日誌群組。如有需要，您可以請求提高配額。

超過日誌串流名稱長度限制

日誌串流名稱具有最大長度限制。當集中化將日誌串流複寫到目的地帳戶時，尾碼會新增至日誌串流名稱。如果產生的日誌串流名稱超過允許的長度上限，則會捨棄記錄，並向客戶帳戶發出`InvalidLogStream`錯誤。

規則運作狀態

在 主控台或使用 `GetCentralizationRuleForOrganization` API 檢查集中化規則的運作狀態。如果規則標示為 UNHEALTHY，請檢閱 `FailureReason` 欄位以取得有關問題的特定詳細資訊。

若要診斷集中化問題，請在 主控台中檢閱集中化規則的運作狀態、檢查 CloudWatch 指標是否有錯誤和限流，以及檢查 AWS CloudTrail 日誌是否有 API 呼叫失敗。如需集中化指標的詳細資訊，請參閱集中化指標和維度。

使用日誌群組和日誌串流

日誌串流是共享相同來源的一系列日誌事件。CloudWatch Logs 中的每個單獨日誌來源組成單獨的日誌串流。

日誌群組是共享相同保留、監控和存取控制設定的日誌串流群組。您可以定義日誌群組，並指定放入每個群組的串流。可以屬於一個日誌群組的日誌串流數量並沒有限制。

對於需要合併多個帳戶和區域的日誌資料的組織，您可以使用 CloudWatch Logs Centralization 自動將日誌群組複寫至中央帳戶。如需詳細資訊，請參閱跨帳戶跨區域日誌集中化。

您可以使用本節中的程序來使用日誌群組和日誌串流。

在 CloudWatch Logs 中建立日誌群組

當您在 Amazon EC2 執行個體上使用《Amazon CloudWatch Logs 使用者指南》先前幾節的步驟安裝 CloudWatch Logs 代理程式時，就已在過程中建立日誌群組。您也可以直接在 CloudWatch 主控台建立日誌群組。

欲建立日誌群組

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌管理。
3. 選擇 Actions (動作)，然後選擇 Create log group (建立日誌群組)。
4. 輸入日誌群組名稱，然後選擇 Create log group (建立日誌群組)。

> **Tip**
>
> 您可以在 CloudWatch 主控台導覽窗格中，在 Favorites and recents (我的最愛和最近的項目) 選單中將日誌群組、儀表板以及警示加入最愛。在 Recently visited (最近造訪) 欄下方，將滑鼠游標停留在要加入最愛的日誌群組上，然後選擇其旁邊的星號。

將日誌傳送到日誌群組

CloudWatch Logs 會自動從數個 AWS 服務接收日誌事件。您也可以使用下列其中一種方法，將其他日誌事件傳送到 CloudWatch Logs：

- CloudWatch 代理程式 - 統一的 CloudWatch 代理程式可以將指標和記錄都傳送至 CloudWatch Logs。如需安裝和使用 CloudWatch 代理程式的相關資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。
- AWS CLI - put-log-events 會將日誌事件批次上傳到 CloudWatch Logs。
- 程式設計 - PutLogEvents API 可讓您以程式設計方式將日誌事件批次上傳到 CloudWatch Logs。

檢視傳送到 CloudWatch Logs 的日誌資料

您可以依串流逐一檢視和捲動透過 CloudWatch Logs 代理程式傳送至 CloudWatch Logs 的日誌資料。您可以檢視指定時間範圍的日誌資料。

欲查看日誌資料

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌管理。
3. 針對 Log Groups (日誌群組)，選擇日誌群組以檢視串流。
4. 在日誌群組清單中，選擇您要檢視的日誌群組清單。
5. 在日誌串流清單中，選擇您要檢視的日誌串流名稱。
6. 若要變更日誌資料的顯示方式，請執行以下其中一項：
   - 若要展開單一日誌事件，選擇日誌事件旁的箭頭。
   - 若要展開所有日誌事件並以純文字檢視，請在日誌事件清單上方選擇 Text (文字)。
   - 若要篩選日誌事件，在搜尋欄位中輸入所需的搜尋篩選條件。如需詳細資訊，請參閱使用篩選條件從日誌事件建立指標。
   - 若要檢視指定日期和時間範圍內的日誌資料，請在搜尋篩選條件旁，選擇日期和時間旁的箭頭。若要指定日期和時間範圍，請選擇 Absolute (絕對)。若要選擇預先定義的分鐘數、小時數、天數或週數，請選擇 Relative (相對)。您也可以在 UTC 和 Local timezone (本機時區) 間進行切換。

使用篩選條件模式搜尋日誌資料

您可以使用用於指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法來搜尋日誌資料。您可以搜尋日誌群組中的所有日誌串流，也可以使用 AWS CLI 搜尋特定的日誌串流。每個搜尋執行時，它會傳回最多找到的第一個資料頁面，以及可擷取下一個頁面資料或繼續搜尋的字符。如果沒有傳回結果，您可以繼續搜尋。

您可以設定您想要查詢的時間範圍，以限制搜尋的範圍。您可以先從較大範圍開始，以查看您有興趣記錄分類的日誌行，然後縮短時間範圍以將視圖範圍限制在您感興趣的時間範圍之日誌。

您也可以直接從您的日誌擷取的指標轉換到對應的日誌。

如果您登入的帳戶設定為 CloudWatch 跨帳戶觀察功能中的監控帳戶，則可以對連結至此監控帳戶之來源帳戶中的日誌事件進行搜尋和篩選。如需詳細資訊，請參閱 CloudWatch 跨帳戶觀察功能。

使用主控台搜尋日誌項目

您可以使用主控台搜尋與指定條件相符的日誌項目。

若要使用主控台搜尋日誌

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌管理。
3. 針對 Log Groups (日誌群組)，輸入包含要搜尋之日誌串流的日誌群組名稱。
4. 對於 Log Streams (日誌串流)，選擇要搜尋的日誌串流名稱。
5. 在 Log events (日誌事件) 下方，輸入要使用的篩選條件語法。

若要使用主控台搜尋某時間範圍內的所有日誌項目

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌管理。
3. 針對 Log Groups (日誌群組)，輸入包含要搜尋之日誌串流的日誌群組名稱。
4. 選擇 Search log group (搜尋日誌群組)。
5. 針對 Log events (日誌事件)，選取日期和時間範圍，然後輸入篩選條件語法。

使用 搜尋日誌項目 AWS CLI

您可以使用 搜尋符合指定條件的日誌項目 AWS CLI。

使用 搜尋日誌項目 AWS CLI

在命令提示中，執行下列 filter-log-events 命令。使用 `--filter-pattern`，將結果限制在指定的篩選條件模式，並使用 `--log-stream-names`，將結果限制在指定的日誌串流。

```
aws logs filter-log-events --log-group-name my-group [--log-stream-
names LIST_OF_STREAMS_TO_SEARCH] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

使用 搜尋指定時間範圍內的日誌項目 AWS CLI

在命令提示中，執行下列 filter-log-events 命令：

```
aws logs filter-log-events --log-group-name my-group [--log-stream-
names LIST_OF_STREAMS_TO_SEARCH] [--start-time 1482197400000] [--end-
time 1482217558365] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

從指標轉換到日誌

您可以從主控台的其他部分取得特定的日誌項目。

若要從儀表板 widget 取得日誌

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Dashboards (儀表板)。
3. 選擇儀表板。
4. 在 widget 中，選擇 View logs (檢視日誌) 圖示，然後選擇 View logs in this time range (查看在這個時間範圍內的日誌)。如果有多個指標篩選條件，從清單中選取一個。如果指標篩選條件的數量多於我們可在清單中顯示的數量，選擇 More metric filters (更多指標篩選條件)，然後選取或搜尋指標篩選條件。

從指標取得到日誌

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇指標。
3. 在搜尋欄位中 All metrics (所有指標) 索引標籤上，輸入指標的名稱，然後按 Enter 鍵。
4. 從搜尋結果選取一或多個指標。
5. 選擇 Actions (動作)、View logs (查看日誌)。如果有多個指標篩選條件，從清單中選取一個。如果指標篩選條件的數量多於我們可在清單中顯示的數量，選擇 More metric filters (更多指標篩選條件)，然後選取或搜尋指標篩選條件。

疑難排解

Search takes too long to complete (需要很長的時間才能完成搜尋)

如果您有許多日誌資料，搜尋可能需要很長的時間來完成。若要加速搜尋，您可以執行以下操作：

- 如果您使用的是 AWS CLI，您可以將搜尋限制為您感興趣的日誌串流。例如，如果您的日誌群組有 1000 個日誌串流，但您只想查看三個您知道相關的日誌串流，您可以使用 AWS CLI 將搜尋限制在日誌群組內的這三個日誌串流。
- 使用較短、更精細的時間範圍，這可減少搜尋的資料量，和加速查詢。

變更 CloudWatch Logs 中的日誌資料保留

在預設情況下，日誌資料會無限期存放於 CloudWatch Logs。不過，您可以設定要將日誌群組中的日誌資料存放多久時間。超過目前保留期間設定的任何資料都會被刪除。您可隨時變更每一群組的日誌保留期間。

> **Note**
>
> CloudWatch Logs 當日誌事件達到保留設定時， 不會立即刪除日誌事件。通常需要經過長達 72 小時才會刪除日誌事件，但在極少數情況下可能需要更長的時間。
> 這表示，如果您將日誌群組變更為在包含超過到期日但尚未實際刪除的日誌事件時具有較長的保留設定，則在到達新的保留日期之後，這些日誌事件最多需要 72 小時才會刪除。若要確保永久刪除日誌資料，請將日誌群組保持為較低的保留設定，直到先前的保留期間結束後 72 小時為止，或者您確認已刪除舊版的日誌事件為止。
> 當日誌事件達到其保留設定時，系統會將其標示以供刪除。經標示以供刪除後，即使之後並未真正刪除，它們也不會再加入您的封存儲存成本。當您使用 API 來擷取 `storedBytes` 值來查看日誌群組正在儲存多少位元組時，這些經標示以供刪除的日誌事件不會納入其中。

若要變更日誌保留期間設定

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。
3. 尋找日誌群組以更新。
4. 在該日誌群組的保留欄中，選擇目前的保留設定，例如永不過期。
5. 在保留設定中，針對之後過期的事件，選擇日誌保留值，然後選擇儲存。

保護日誌群組免於刪除

您可以選擇性地啟用刪除保護，以防止意外刪除重要的日誌群組。如需刪除保護的詳細資訊，請參閱保護日誌群組免於刪除。

保護日誌群組免於刪除

啟用刪除保護

您可以在建立新日誌群組或在現有日誌群組上啟用刪除保護。在建立日誌群組期間，選取「啟用刪除保護」或傳遞 參數 `--deletion-protection-enabled`。根據預設，不會啟用刪除保護。

啟用或停用現有日誌群組的刪除保護（主控台）

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇日誌管理。
3. 選取您要保護的日誌群組。
4. 選擇動作、編輯刪除保護。
5. 在對話方塊中，檢閱並提交變更。

如果使用 AWS CLI，若要在現有日誌群組上啟用刪除保護：

```
aws logs put-log-group-deletion-protection \
--log-group-identifier "/my-application/logs" \
--deletion-protection-enabled
```

若要移除現有日誌群組的刪除保護：

```
aws logs put-log-group-deletion-protection \
--log-group-identifier "/my-application/logs" \
--no-deletion-protection-enabled
```

錯誤處理

如果您嘗試刪除已啟用刪除保護的日誌群組，您會收到`ValidationException`訊息為「無法刪除已啟用刪除保護的日誌群組。請先停用刪除保護。」

在 Amazon CloudWatch Logs 中標記日誌群組

您可以將自己的中繼資料以標籤形式，指派到您在 Amazon CloudWatch Logs 中建立的日誌群組。標籤是您為日誌群組定義的鍵值對。使用標籤是一種簡單但強大的方法來管理 AWS 資源和組織資料，包括帳單資料。

> **Note**
>
> 您可以使用標籤來控制對 CloudWatch Logs 資源的存取，包括對日誌群組和目的地的存取。由於日誌群組與日誌串流之間的階層關係，系統會在日誌群組層級控制對日誌串流的存取。如需有關使用標籤來控制存取的詳細資訊，請參閱使用標籤控制對 Amazon Web Services 資源的存取。

目錄

標籤基本概念

您可以使用 AWS CloudFormation AWS CLI或 CloudWatch Logs API 來完成下列任務：

- 當您建立日誌群組時為其新增標籤。
- 新增標籤到現有的日誌群組。
- 列出日誌群組的標籤。
- 從日誌群組移除標籤。

您可以使用標籤來分類您的日誌群組。例如，您可以依用途、擁有者或環境來為它們分類。由於您定義了每個標籤的鍵和值，您可以建立一組自訂的類別，以符合您的特定需求。例如，您可以定義一組標籤，可協助您依照日誌群組的擁有者和關聯的應用程式來追蹤日誌群組。以下是數個標籤的範例：

- 專案：專案名稱

- 擁有者：名稱
- 用途：負載測試
- 應用程式：應用程式名稱
- 環境：生產

使用標記追蹤成本

您可以使用標籤來分類和追蹤您的 AWS 成本。當您將標籤套用至 AWS 資源時，包括日誌群組，AWS 成本分配報告會包含依標籤彙總的用量和成本。您可以套用代表業務類別 (例如成本中心、應用程式名稱或擁有者) 的標籤，來整理多個服務中的成本。如需詳細資訊，請參閱《AWS Billing 使用者指南》中的將成本分配標籤用於自訂帳單報告。

標籤限制

下列限制適用於標籤。

基本限制

- 每個日誌群組的標籤數上限為 50。
- 標籤鍵與值皆區分大小寫。
- 您無法變更或編輯已刪除日誌群組的標籤。

標籤鍵限制

- 每個標籤鍵都必須是唯一的。如果您新增具有已使用索引鍵的標籤，則新的標籤會覆寫現有鍵值對。
- 您無法以 啟動標籤金鑰，`aws:`因為此字首保留給 使用 AWS。 會代表您 AWS 建立以此字首開頭的標籤，但您無法編輯或刪除它們。
- 標籤鍵的長度必須介於 1 到 128 個 Unicode 字元之間。
- 標籤鍵必須包含下列字元：Unicode 字母、數字、空格以及下列特殊字元：`_ . / = + - @`。

標籤值限制

- 標籤值的長度必須介於 0 到 255 個 Unicode 字元之間。
- 標籤值可以空白。否則，它們必須包含下列字元：Unicode 字母、數字、空格以及下列任何特殊字元：`_ . / = + - @`。

使用 標記日誌群組 AWS CLI

您可以使用 AWS CLI來新增、列出和移除標籤。如需範例，請參閱下列文件：

create-log-group

建立一個日誌群組。當您建立日誌群組時，您可以選擇性新增標籤。

tag-resource

將一或多個標籤 (鍵/值對) 指派給指定的 CloudWatch Logs 資源。

list-tags-for-resource

顯示與 CloudWatch Logs 資源相關聯的標籤。

untag-resource

從指定的 CloudWatch Logs 資源移除一或多個標籤。

使用 CloudWatch Logs API 標記日誌群組

您可以使用 CloudWatch Logs API 來新增、列出和移除標籤。如需範例，請參閱下列文件：

CreateLogGroup

建立一個日誌群組。當您建立日誌群組時，您可以選擇性新增標籤。

TagResource

將一或多個標籤 (鍵/值對) 指派給指定的 CloudWatch Logs 資源。

ListTagsForResource

顯示與 CloudWatch Logs 資源相關聯的標籤。

UntagResource

從指定的 CloudWatch Logs 資源移除一或多個標籤。

使用 在 CloudWatch Logs 中加密日誌資料 AWS Key Management Service

在 CloudWatch Logs 中，日誌群組資料一律會加密。根據預設，CloudWatch Logs 會使用伺服器端加密搭配 256 位元進階加密標準 Galois/計數器模式 (AES-GCM) 來加密靜態日誌資料。您也可以使用

AWS Key Management Service 進行此加密。如果您這麼做，則會使用 AWS KMS 金鑰完成加密。AWS KMS 使用 在日誌群組層級啟用加密，方法是在您建立日誌群組時或在日誌群組存在之後，將 KMS 金鑰與日誌群組建立關聯。

> **Important**
>
> CloudWatch Logs 現在支援加密內容，用 `kms:EncryptionContext:aws:logs:arn` 做為金鑰，而日誌群組的 ARN 做為該金鑰的值。如果您有日誌群組已使用 KMS 加密，並希望能限制金鑰與單一帳戶和日誌群組搭配使用，您應該在 IAM 政策中指派包含條件的新 KMS 金鑰。如需詳細資訊，請參閱AWS KMS 金鑰和加密內容。

> **Important**
>
> CloudWatch Logs 現在支援`kms:ViaService`允許日誌代表您進行 AWS KMS 呼叫。您應該將此新增至在金鑰政策或 IAM 中呼叫 CloudWatch Logs 的角色。如需詳細資訊，請參閱 kms:ViaService。

建立 KMS 金鑰與日誌群組的關聯後，針對該日誌群組新擷取的所有資料，就會使用此金鑰加密。此資料在整個保留期間都以加密的格式儲存。每當請求此資料時，CloudWatch Logs 會解密此資料。每當有人請求取得加密的資料時，CloudWatch Logs 必須擁有 KMS 金鑰許可。

如果您之後取消 KMS 金鑰與日誌群組的關聯，CloudWatch Logs 會使用 CloudWatch Logs 預設的加密方法來加密新擷取的資料。先前使用 KMS 金鑰加密的所有擷取資料仍會使用 KMS 金鑰加密。由於 CloudWatch Logs 仍可繼續參照該金鑰，因此依然可在 KMS 金鑰解除關聯後傳回那些資料。但若稍後將金鑰停用，則 CloudWatch Logs 將無法讀取使用該金鑰加密的日誌。

> **Important**
>
> CloudWatch Logs 僅支援對稱 KMS 金鑰。請勿使用非對稱金鑰來加密日誌群組中的資料。如需詳細資訊，請參閱使用對稱和非對稱金鑰。

限制

- 若要執行下列步驟，您必須擁有下列許可：`kms:CreateKey`、`kms:GetKeyPolicy` 和 `kms:PutKeyPolicy`。

- 在您建立或取消金鑰與日誌群組的關聯後，操作將在 5 分鐘內生效。
- 如果您撤銷 CloudWatch Logs 對已關聯金鑰的存取權，或刪除所關聯的 KMS 金鑰，您將無法再擷取 CloudWatch Logs 中的加密資料。
- 您無法使用 CloudWatch 主控台將 KMS 金鑰與現有日誌群組建立關聯。

步驟 1：建立 AWS KMS 金鑰

若要建立 KMS 金鑰，請使用以下 create-key 命令：

```
aws kms create-key
```

輸出包含金鑰 ID 和金鑰的 Amazon Resource Name (ARN)。下列為範例輸出：

```
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1478910250.94,
        "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-
e40cb0d29f59",
        "AWSAccountId": "123456789012",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

步驟 2：設定 KMS 金鑰許可

根據預設，所有 AWS KMS 金鑰都是私有的。只有資源擁有者可以使用它來加密和解密資料。然而，資源擁有者可以授與其他使用者和資源存取 KMS 金鑰的許可。在此步驟中，您會授予 CloudWatch Logs 服務主體和呼叫者角色使用金鑰的許可。此服務主體必須位於存放 KMS 金鑰的相同 AWS 區域中。

根據最佳實務，建議您將 KMS 金鑰的使用限制為您指定的 AWS 帳戶或日誌群組。

首先，使用以下 get-key-policy 命令，將 KMS 金鑰的預設政策儲存為 policy.json：

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

在文字編輯器中開啟 policy.json 檔案，並從下列其中一個陳述式中加入區段 (以粗體顯示)。使用逗號從新陳述式中分隔現有陳述式。這些陳述式使用Condition區段來增強 AWS KMS 金鑰的安全性。如需詳細資訊，請參閱AWS KMS 金鑰和加密內容。

此範例中的 Condition 區段會將金鑰限制為單一日誌群組 ARN。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:Describe*"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "ArnEquals": {
                "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-
east-1:111122223333:log-group:log-group-name"
                }
            }
        }
    ]
}
```

本範例中的 Condition 區段將 AWS KMS 金鑰限用於指定的帳戶，但可用於任何日誌群組。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:Describe*"
            ],
            "Resource": "*",
            "Condition": {
                "ArnLike": {
```

```
                "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-
east-1:123456789012:*"
                }
            }
        }
    ]
}
```

接著，將許可新增至將呼叫 CloudWatch Logs 的角色。您可以將其他陳述式新增至 AWS KMS 金鑰政策，或透過角色本身的 IAM 來執行此操作。CloudWatch Logs 會使用 代表客戶`kms:ViaService`呼叫 AWS KMS 。如需詳細資訊，請參閱 kms:ViaService。

若要在 AWS KMS 金鑰政策中新增許可，請將下列其他陳述式新增至您的金鑰政策。如果您使用此方法作為最佳實務，請將政策範圍限定為將與 AWS KMS 加密日誌群組互動的角色。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account_id:role/role_name"
  },
  "Action": [
    "kms:Encrypt",
    "kms:ReEncrypt*",
    "kms:Decrypt",
    "kms:GenerateDataKey*",
    "kms:Describe*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "logs.region.amazonaws.com"
      ]
    }
  }
}
```

或者，如果您想要在 IAM 中管理角色許可，您可以透過下列政策新增同等許可。這可以新增到現有的角色政策，或做為額外的個別政策連接到角色。如果您使用此方法作為最佳實務，請將政策範圍限定為僅用於日誌加密的 AWS KMS 金鑰。如需詳細資訊，請參閱編輯 IAM 政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:ReEncrypt*",
                "kms:Decrypt",
                "kms:GenerateDataKey*",
                "kms:Describe*"
            ],
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": [
                        "logs.us-east-1.amazonaws.com"
                    ]
                }
            },
            "Resource": "arn:aws:kms:us-east-1:444455556666:key/key_id"
        }
    ]
}
```

最後，使用下列 put-key-policy 命令新增更新的政策：

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

步驟 3：為 KMS 金鑰與日誌群組建立關聯

您可以在建立日誌群組時或建立完成後，為 KMS 金鑰與日誌群組建立關聯。

如果要確認日誌群組是否已經有相關聯的 KMS 金鑰，請使用以下 describe-log-groups 命令：

```
aws logs describe-log-groups --log-group-name-prefix "log-group-name-prefix"
```

如果輸出包含 kmsKeyId 欄位，則日誌群組會與該欄位值所顯示的索引鍵相關聯。

若要在建立日誌群組時與 KMS 金鑰建立關聯

使用 create-log-group 命令，如下所示：

```
aws logs create-log-group --log-group-name my-log-group --kms-key-id "key-arn"
```

若要為 KMS 金鑰與現有的日誌群組建立關聯

使用 associate-kms-key 命令，如下所示：

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id "key-arn"
```

步驟 4：取消金鑰與日誌群組的關聯

若要取消 KMS 金鑰與日誌群組的關聯，請使用以下 disassociate-kms-key 命令：

```
aws logs disassociate-kms-key --log-group-name my-log-group
```

AWS KMS 金鑰和加密內容

為了增強 AWS Key Management Service 金鑰和加密日誌群組的安全性，CloudWatch Logs 現在會將日誌群組 ARNs 做為用於加密日誌資料之加密內容的一部分。加密內容是一組做為額外驗證資料的索引鍵/值組。加密內容可讓您使用 IAM 政策條件，依 AWS 帳戶和日誌群組限制對 AWS KMS 金鑰的存取。如需詳細資訊，請參閱加密內容和 IAM JSON 政策元素：Condition。

建議您針對每個加密的日誌群組使用不同的 KMS 金鑰。

如果您有先前加密的日誌群組，但現在希望將日誌群組變更為使用只適用於該日誌群組的新 KMS 金鑰，請依照下列步驟執行。

透過政策將金鑰設為特定日誌群組專用，將加密日誌群組轉換為使用 KMS 金鑰

1. 請輸入下列指令，以尋找日誌群組目前金鑰的 ARN：

   ```
   aws logs describe-log-groups
   ```

輸出包括這一行。記下 ARN。步驟 7 中會用到。

```
...
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/01234567-89ab-
cdef-0123-456789abcdef"
...
```

2. 輸入以下命令來建立新的 KMS 金鑰：

```
aws kms create-key
```

3. 輸入下列命令，將新金鑰的政策儲存至 policy.json 檔案：

```
aws kms get-key-policy --key-id new-key-id --policy-name default --output text > ./
policy.json
```

4. 使用文字編輯器來開啟 policy.json，並將 Condition 運算式新增至政策：

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
```

```
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:Describe*"
            ],
            "Resource": "*",
            "Condition": {
                "ArnLike": {
                "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-
east-1:111122223333:log-group:LOG-GROUP-NAME"
                }
            }
        }
    ]
}
```

5. 輸入下列命令，將更新的政策新增至新的 KMS 金鑰：

```
aws kms put-key-policy --key-id new-key-ARN --policy-name default --policy file://
policy.json
```

6. 輸入下列命令，將政策與日誌群組建立關聯：

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id new-key-ARN
```

CloudWatch Logs 現在會使用新金鑰來加密所有新資料。

7. 接下來，撤銷舊金鑰的所有權限，除了 `Decrypt` 以外。首先，輸入下列命令以擷取舊政策：

```
aws kms get-key-policy --key-id old-key-ARN --policy-name default --output text
 > ./policy.json
```

8. 使用文字編輯器來開啟 `policy.json`，並移除 `Action` 清單中的所有值，除了 `kms:Decrypt` 以外

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
```

```
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.region.amazonaws.com"
            },
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

9. 輸入下列命令，將更新的政策新增至舊金鑰：

```
aws kms put-key-policy --key-id old-key-ARN --policy-name default --policy file://
policy.json
```

使用遮罩功能協助保護敏感日誌資料

您可以使用日誌群組資料保護政策，協助保護 CloudWatch Logs 擷取的敏感資料。這些政策可讓您稽核和遮罩出現在帳戶中日誌群組擷取之日誌事件中的敏感資料。

建立資料保護政策後，預設為會在所有出口點遮罩與您選取的資料識別符相符的敏感資料，包括 CloudWatch Logs Insights、指標篩選條件和訂閱篩選條件。只有具有 `logs:Unmask` IAM 許可的使用者才能檢視未遮罩的資料。

您可以為帳戶中的所有日誌群組建立資料保護政策，也可以為個別日誌群組建立資料保護政策。當您為整個帳戶建立政策時，它會套用至現有的日誌群組和未來建立的日誌群組。

如果您為整個帳戶建立資料保護政策，並且也為單一日誌群組建立政策，則這兩個政策都會套用至該日誌群組。在任一政策中指定的所有受管資料識別符都會在該日誌群組中進行稽核和遮罩。

> **Note**
>
> 標準和不常存取日誌類別中的日誌群組都支援遮罩敏感資料。如需日誌類別的詳細資訊，請參閱 日誌類別。

每個日誌群組只能有一個日誌群組層級資料保護政策，但該政策可以指定許多受管資料識別符來稽核和遮罩。資料保護政策的限制為 30,720 個字元。

> **Important**
>
> 將敏感資料擷取至日誌群組時，系統會偵測這些資料並加以遮罩。系統不會遮罩在您設定資料保護政策之前擷取至日誌群組的日誌事件。

CloudWatch Logs 支援許多受管資料識別符，提供預先設定的資料類型，您可以選擇這些資料類型來保護財務資料、個人健康資訊 (PHI) 和個人身分識別資訊 (PII)。CloudWatch Logs 資料保護功能可讓您利用模式比對和機器學習模型來偵測敏感資料。對於某些類型的受管資料識別符，偵測也取決於尋找與敏感資料相鄰的特定關鍵字。您也可以使用自訂資料識別符來建立針對特定使用案例量身打造的資料識別符。

當偵測到符合您所選取資料識別碼的敏感資料時，就會向 CloudWatch 發出指標。這是 LogEventsWithFindings 指標，其在 AWS/Logs 命名空間中發出。您可以使用此指標建立 CloudWatch 警示，並且可以在圖形和儀表板中將其視覺化。資料保護發出的指標是付費指標，但在此免費提供。如需有關 CloudWatch Logs 傳送至 CloudWatch 之指標的詳細資訊，請參閱 使用 CloudWatch 指標監控使用量。

每個受管資料識別符旨在偵測特定類型的敏感資料，例如特定國家或地區的信用卡號碼、 AWS 私密存取金鑰或護照號碼。建立資料保護政策時，您可以將 CloudWatch Logs 設定為使用這些識別符，來分析由日誌群組擷取的日誌，並在偵測到日誌時採取動作。

CloudWatch Logs 資料保護功能可以使用受管資料識別符，偵測下列類別的敏感資料：

- 登入資料，例如私有金鑰或 AWS 私密存取金鑰
- 財務資訊，例如信用卡號碼。
- 個人身分識別資訊 (PII)，例如駕照或社會安全號碼
- 受保護醫療資訊 (PHI)，例如健康保險或醫療識別號碼
- 裝置識別符，例如 IP 地址或 MAC 地址

如需有關可保護之資料類型的詳細資訊，請參閱您可以保護的資料類型。

內容

了解資料保護政策

主題

什麼是資料保護政策？

CloudWatch Logs 使用資料保護政策來選取您要掃描的敏感資料，以及您想要執行以保護資料的動作。若要選擇感興趣的敏感資料，請使用資料識別符。然後，CloudWatch Logs 資料資料保護會使用機器學習和模式比對來偵測敏感資料。若要根據找到的資料識別符採取行動，您可以定義稽核和去識別化操作。這些操作可讓您記錄找到 (或未找到) 的敏感資料，並在檢視日誌事件時遮罩敏感資料。

資料保護政策的結構如何？

如下圖所示，資料保護政策文件包含以下元素：

- 在文件最上方選用的整體政策資訊
- 定義稽核和去識別動作的一條陳述式

每個 CloudWatch Logs 日誌群組只能定義一種資料保護政策。資料保護政策可以有一或多個拒絕或去識別化陳述式，但只能有一個稽核陳述式。

資料保護政策的 JSON 屬性

資料保護政策需要下列基本政策資訊才能識別：

- Name - 政策名稱。
- Description (選用) - 政策描述。
- Version - 政策語言版本。目前版本是 2021-06-01。
- Statement - 指定資料保護政策動作的陳述式清單。

```
{
  "Name": "CloudWatchLogs-PersonalInformation-Protection",
  "Description": "Protect basic types of sensitive data",
  "Version": "2021-06-01",
  "Statement": [
        ...
  ]
}
```

政策陳述式的 JSON 屬性

政策陳述式會設定資料保護操作的偵測內容。

- Sid (選用) - 陳述式識別符。
- DataIdentifier – CloudWatch Logs 應掃描的敏感資料。例如，姓名、地址或電話號碼。
- 操作 – 后續動作 (稽核或去識別)。CloudWatch Logs 會在找到敏感資料時執行這些動作。

```
{
  ...
  "Statement": [
    {
      "Sid": "audit-policy",
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/Address"
      ],
      "Operation": {
        "Audit": {
```

```
        "FindingsDestination": {}
      }
    }
  },
```

政策陳述式操作的 JSON 屬性

政策陳述式會設定下列其中一項資料保護操作。

- 稽核 – 發出指標和問題清單報告，而不會中斷日誌記錄。符合的字串會增加 LogEventsWithFindings 指標，CloudWatch Logs 將此指標發佈到 CloudWatch 中的 AWS/Logs 命名空間。您可以使用這些指標建立警示。

  如需問題清單報告的範例，請參閱 稽核問題清單報告。

  如需有關 CloudWatch Logs 傳送至 CloudWatch 之指標的詳細資訊，請參閱 使用 CloudWatch 指標監控使用量。
- 去識別 – 遮罩敏感資料，而不會中斷日誌記錄。

必須具備 IAM 許可才能建立或使用資料保護政策

若要能夠使用日誌群組的資料保護政策，您必須具有下表所示的特定許可。帳戶層級的資料保護政策和套用至單一日誌群組的資料保護政策的許可有所不同。

帳戶層級資料保護政策所需的許可

> **Note**
>
> 如果您要在 Lambda 函數內執行這些作業，Lambda 執行角色和許可界限也必須包含下列許可。

作業	需要 IAM 許可	資源
建立不具有稽核目的地的資料保護政策	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*

作業	需要 IAM 許可	資源
建立以 CloudWatch Logs 為稽核目的地的資料保護政策	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*
	`logs:PutResourcePolicy`	*
	`logs:DescribeResourcePolicies`	*
	`logs:DescribeLogGroups`	*
使用 Firehose 作為稽核目的地來建立資料保護政策	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*
	`firehose:TagDeliveryStream`	`arn:aws:logs:::deliverystream/` *YOUR_DELIVERY_STREAM*
建立以 Amazon S3 為稽核目的地的資料保護政策	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*

作業	需要 IAM 許可	資源
	logs:CreateLogDelivery	*
	s3:GetBucketPolicy	arn:aws:s3::: *YOUR_BUCKET*
	s3:PutBucketPolicy	arn:aws:s3::: *YOUR_BUCKET*
取消遮罩指定日誌群組中的遮罩日誌事件	logs:Unmask	arn:aws:logs:::log-group:*
檢視現有的資料保護政策	logs:GetDataProtectionPolicy	*
刪除資料保護政策	logs:DeleteAccountPolicy	*
	logs:DeleteDataProtectionPolicy	*

如果有任何資料保護稽核日誌已傳送至目的地，則也將日誌傳送至相同目的地的其他策略僅需要 `logs:PutDataProtectionPolicy` 和 `logs:CreateLogDelivery` 許可。

單一日誌群組之資料保護政策所需的許可

> **Note**
>
> 如果您要在 Lambda 函數內執行這些作業，Lambda 執行角色和許可界限也必須包含下列許可。

作業	需要 IAM 許可	資源
建立不具有稽核目的地的資料保護政策	logs:PutDataProtectionPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :*
建立以 CloudWatch Logs 為稽核目的地的資料保護政策	logs:PutDataProtectionPolicy logs:CreateLogDelivery logs:PutResourcePolicy logs:DescribeResourcePolicies logs:DescribeLogGroups	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * * * *
使用 Firehose 作為稽核目的地來建立資料保護政策	logs:PutDataProtectionPolicy logs:CreateLogDelivery firehose:TagDeliveryStream	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * arn:aws:logs:::deliverystream/ *YOUR_DELIVERY_STREAM*
建立以 Amazon S3 為稽核目的地的資料保護政策	logs:PutDataProtectionPolicy logs:CreateLogDelivery s3:GetBucketPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * arn:aws:s3::: *YOUR_BUCKET*

作業	需要 IAM 許可	資源
	s3:PutBucketPolicy	arn:aws:s3::: *YOUR_BUCKET*
取消遮罩已遮罩的日誌事件	logs:Unmask	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :*
檢視現有的資料保護政策	logs:GetDataProtectionPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :*
刪除資料保護政策	logs:DeleteDataProtectionPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :*

如果有任何資料保護稽核日誌已傳送至目的地，則也將日誌傳送至相同目的地的其他策略僅需要 logs:PutDataProtectionPolicy 和 logs:CreateLogDelivery 許可。

資料保護政策範例

下列政策範例可讓使用者建立、檢視及刪除資料保護政策，這些政策可將稽核調查結果傳送至全部三種稽核目的地類型。它不允許使用者檢視未遮罩的資料。

JSON

```
{
"Version":"2012-10-17",
"Statement": [
    {
        "Sid": "AllowLogDeliveryConfiguration",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogDelivery",
            "logs:PutResourcePolicy",
            "logs:DescribeLogGroups",
            "logs:DescribeResourcePolicies"
        ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "AllowDataProtectionAndBucketConfiguration",
            "Effect": "Allow",
            "Action": [
                "logs:GetDataProtectionPolicy",
                "logs:DeleteDataProtectionPolicy",
                "logs:PutDataProtectionPolicy",
                "s3:PutBucketPolicy",
                "firehose:TagDeliveryStream",
                "s3:GetBucketPolicy"
            ],
            "Resource": [
            "arn:aws:firehose:us-east-1:111122223333:deliverystream/delivery-
stream-name",
            "arn:aws:s3:::amzn-s3-demo-destination-bucket",
            "arn:aws:logs:us-east-1:111122223333:log-group:log-group-name:*"
            ]
        }
    ]
}
```

建立帳戶層級資料保護政策

您可以使用 CloudWatch Logs 主控台或 AWS CLI 命令來建立資料保護政策，以遮罩帳戶中所有日誌群組的敏感資料。這樣做會影響目前的日誌群組和您未來建立的日誌群組。

> **Important**
>
> 將敏感資料擷取至日誌群組時，系統會偵測這些資料並加以遮罩。系統不會遮罩在您設定資料保護政策之前擷取至日誌群組的日誌事件。

主題

- 主控台
- AWS CLI

主控台

使用主控台建立帳戶層級資料保護政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Settings (設定)。它位於清單底部附近。
3. 選擇 Logs (日誌) 索引標籤。
4. 選擇設定。
5. 針對受管資料識別符，選取您要為所有日誌群組稽核和遮罩的資料類型。您可以在選取方塊中輸入內容以尋找所需的識別符。

   我們建議您只選取與日誌資料和業務相關的資料識別符。選擇的資料類型過多可能會導致誤報。

   如需有關可保護的資料類型的詳細資訊，請參閱您可以保護的資料類型。
6. （選用）如果您想要使用自訂資料識別符稽核和遮罩其他類型的資料，請選擇新增自訂資料識別符。然後輸入資料類型的名稱，以及用來在日誌事件中搜尋該資料類型的規則表達式。如需詳細資訊，請參閱自訂資料識別符。

   單一資料保護政策最多可包含 10 個自訂資料識別符。每個定義自訂資料識別符的規則表達式都必須為 200 個字元或更少。
7. (選用) 選擇向其傳送稽核問題清單的一或多個服務。即使您選擇不將稽核問題清單傳送至任何這些服務，系統仍然會遮罩您選取的敏感資料類型。
8. 選擇 Activate data protection (啟動資料保護)。

AWS CLI

使用 AWS CLI 建立資料保護政策

1. 使用文字編輯器來建立名為 `DataProtectionPolicy.json` 的政策檔案。如需有關政策語法的資訊，請參閱下一節。
2. 輸入以下命令：

```
aws logs put-account-policy \
--policy-name TEST_POLICY --policy-type "DATA_PROTECTION_POLICY" \
--policy-document file://policy.json \
--scope "ALL" \
--region us-west-2
```

AWS CLI 或 API 操作的資料保護政策語法

當您建立要在 AWS CLI 命令或 API 操作中使用的 JSON 資料保護政策時，該政策必須包含兩個 JSON 區塊：

- 第一個區塊必須同時包含 DataIdentifer 陣列和具有 Audit 動作的 Operation 屬性。DataIdentifer 陣列會列出您要遮罩的敏感資料類型。如需所有可用選項的詳細資訊，請參閱您可以保護的資料類型。

  具有 Audit 動作的 Operation 屬性為必要項目，如此才能找到敏感資料術語。此 Audit 動作必須包含 FindingsDestination 物件。您可以選擇使用此 FindingsDestination 物件，來列出要向其傳送稽核問題清單報告的一或多個目的地。如果您指定目標，例如日誌群組、Amazon Data Firehose 串流和 S3 儲存貯體，它們必須已存在。如需稽核問題清單報告的範例，請參閱 稽核問題清單報告。
- 第二個區塊必須同時包含 DataIdentifer 陣列和具有 Deidentify 動作的 Operation 屬性。DataIdentifer 陣列必須與政策第一個區塊中的 DataIdentifer 陣列完全一致。

  具有 Deidentify 動作的 Operation 屬性用於實際遮罩資料，其必須包含 "MaskConfig": {} 物件。 "MaskConfig": {} 物件必須是空的。

以下是僅使用受管資料識別符的資料保護政策範例。此政策會遮罩電子郵件地址和美國駕照。

如需指定自訂資料識別符的政策相關資訊，請參閱 在您的資料保護政策中使用自訂資料識別符。

```
{
    "Name": "data-protection-policy",
    "Description": "test description",
    "Version": "2021-06-01",
    "Statement": [{
            "Sid": "audit-policy",
            "DataIdentifier": [
                "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
                "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
            ],
            "Operation": {
                "Audit": {
                    "FindingsDestination": {
                        "CloudWatchLogs": {
                            "LogGroup": "EXISTING_LOG_GROUP_IN_YOUR_ACCOUNT,"
                        },
                        "Firehose": {
```

```
                        "DeliveryStream": "EXISTING_STREAM_IN_YOUR_ACCOUNT"
                    },
                    "S3": {
                        "Bucket": "EXISTING_BUCKET"
                    }
                }
            }
        }
    },
    {
        "Sid": "redact-policy",
        "DataIdentifier": [
            "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
            "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
        ],
        "Operation": {
            "Deidentify": {
                "MaskConfig": {}
            }
        }
    }
  ]
}
```

建立單一日誌群組的資料保護政策

您可以使用 CloudWatch Logs 主控台或 AWS CLI 命令建立資料保護政策來遮罩敏感資料。

您可以為每個日誌群組指派一個資料保護政策。每個資料保護政策都可以稽核多種類型的資訊。每個資料保護政策都可以包含一份稽核聲明。

主題

- 主控台
- AWS CLI

主控台

若要使用主控台建立資料保護政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。

3. 選擇日誌群組的名稱。
4. 選擇 Actions (動作)、Create data protection policy (建立資料保護政策)。
5. 針對受管資料識別符，選取您要在此日誌群組中稽核和遮罩的資料類型。您可以在選取方塊中輸入內容以尋找所需的識別符。

   我們建議您只選取與日誌資料和業務相關的資料識別符。選擇的資料類型過多可能會導致誤報。

   如需您可以使用受管資料識別符保護哪些資料類型的詳細資訊，請參閱 您可以保護的資料類型。
6. （選用）如果您想要使用自訂資料識別符稽核和遮罩其他類型的資料，請選擇新增自訂資料識別符。然後輸入資料類型的名稱，以及用來在日誌事件中搜尋該資料類型的規則表達式。如需詳細資訊，請參閱自訂資料識別符。

   單一資料保護政策最多可包含 10 個自訂資料識別符。每個定義自訂資料識別符的規則表達式都必須為 200 個字元或更少。
7. (選用) 選擇向其傳送稽核問題清單的一或多個服務。即使您選擇不將稽核問題清單傳送至任何這些服務，系統仍然會遮罩您選取的敏感資料類型。
8. 選擇 Activate data protection (啟動資料保護)。

AWS CLI

使用 AWS CLI 建立資料保護政策

1. 使用文字編輯器來建立名為 `DataProtectionPolicy.json` 的政策檔案。如需有關政策語法的資訊，請參閱下一節。
2. 輸入以下命令：

```
aws logs put-data-protection-policy --log-group-identifier "my-log-group" --policy-
document file:///Path/DataProtectionPolicy.json --region us-west-2
```

AWS CLI 或 API 操作的資料保護政策語法

當您建立要在 AWS CLI 命令或 API 操作中使用的 JSON 資料保護政策時，該政策必須包含兩個 JSON 區塊：

- 第一個區塊必須同時包含 `DataIdentifer` 陣列和具有 `Audit` 動作的 `Operation` 屬性。`DataIdentifer` 陣列會列出您要遮罩的敏感資料類型。如需所有可用選項的詳細資訊，請參閱您可以保護的資料類型。

具有 Audit 動作的 Operation 屬性為必要項目，如此才能找到敏感資料術語。此 Audit 動作必須包含 FindingsDestination 物件。您可以選擇使用此 FindingsDestination 物件，來列出要向其傳送稽核問題清單報告的一或多個目的地。如果您指定目標，例如日誌群組、Amazon Data Firehose 串流和 S3 儲存貯體，它們必須已存在。如需稽核問題清單報告的範例，請參閱 稽核問題清單報告。

- 第二個區塊必須同時包含 DataIdentifer 陣列和具有 Deidentify 動作的 Operation 屬性。DataIdentifer 陣列必須與政策第一個區塊中的 DataIdentifer 陣列完全一致。

  具有 Deidentify 動作的 Operation 屬性用於實際遮罩資料，其必須包含 "MaskConfig": {} 物件。 "MaskConfig": {} 物件必須是空的。

以下是遮罩電子郵件地址和美國駕照的資料保護政策範例。

```
{
    "Name": "data-protection-policy",
    "Description": "test description",
    "Version": "2021-06-01",
    "Statement": [{
            "Sid": "audit-policy",
            "DataIdentifier": [
                "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
                "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
            ],
            "Operation": {
                "Audit": {
                    "FindingsDestination": {
                        "CloudWatchLogs": {
                            "LogGroup": "EXISTING_LOG_GROUP_IN_YOUR_ACCOUNT,"
                        },
                        "Firehose": {
                            "DeliveryStream": "EXISTING_STREAM_IN_YOUR_ACCOUNT"
                        },
                        "S3": {
                            "Bucket": "EXISTING_BUCKET"
                        }
                    }
                }
            }
        },
        {
```

```
            "Sid": "redact-policy",
            "DataIdentifier": [
                "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
                "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
            ],
            "Operation": {
                "Deidentify": {
                    "MaskConfig": {}
                }
            }
        }
    ]
}
```

檢視未遮罩的資料

若要檢視未遮罩的資料，使用者必須具有 `logs:Unmask` 許可。具有此許可的使用者可透過以下方式查看未遮罩的資料：

- 檢視日誌串流中的事件時，選擇 Display (顯示)、Unmask (解除遮罩)。
- 使用包含 unmask(@message) 命令的 CloudWatch Logs Insights 查詢。下列範例查詢會以未遮罩的方式顯示串流中最近 20 個日誌事件：

  ```
  fields @timestamp, @message, unmask(@message)
  | sort @timestamp desc
  | limit 20
  ```

  如需有關 CloudWatch Logs Insights 命令的詳細資訊，請參閱 CloudWatch Logs Insights 語言查詢語法。
- 搭配 `unmask` 參數使用 GetLogEvents 或 FilterLogEvents 操作。

CloudWatchLogsFullAccess 政策包含 `logs:Unmask` 許可。若要將 `logs:Unmask` 授予給不具備 CloudWatchLogsFullAccess 許可的使用者，您可以將自訂 IAM 政策附加到該使用者。如需詳細資訊，請參閱新增許可到使用者 (主控台)。

稽核問題清單報告

如果您設定 CloudWatch Logs 資料保護稽核政策，將稽核報告寫入 CloudWatch Logs、Amazon S3 或 Firehose，這些調查結果報告類似於下列範例。CloudWatch Logs 會針對每個包含敏感資料的日誌事件寫入一份問題清單報告。

```
{
    "auditTimestamp": "2023-01-23T21:11:20Z",
    "resourceArn": "arn:aws:logs:us-west-2:111122223333:log-group:/aws/lambda/
MyLogGroup:*",
    "dataIdentifiers": [
        {
            "name": "EmailAddress",
            "count": 2,
            "detections": [
                {
                    "start": 13,
                    "end": 26
                },
{
                    "start": 30,
                    "end": 43
                }
            ]
        }
    ]
}
```

報告中的欄位如下所示：

- `resourceArn` 欄位會顯示在其中找到敏感資料的日誌群組。
- `dataIdentifiers` 物件會顯示您正在稽核的某種敏感資料的問題清單相關資訊。
- `name` 欄位可識別此區段所報告的敏感資料類型。
- `count` 欄位會顯示此種敏感資料在日誌事件中出現的次數。
- `start` 和 `end` 欄位會依字元計數顯示日誌事件中每次出現敏感資料的位置。

上一個範例顯示在一個日誌事件中尋找兩個電子郵件地址的報告。第一個電子郵件地址從日誌事件的第 13 個字元開始，並在第 26 個字元處結束。第二個電子郵件地址從第 30 個字元一直到第 43 個字元。

即使此日誌事件有兩個電子郵件地址，`LogEventsWithFindings` 指標的值也只會遞增 1，因為該指標會對包含敏感資料的日誌事件計數，而非計算敏感資料的出現次數。

將稽核問題清單傳送至受 保護的 儲存貯體所需的金鑰政策 AWS KMS

透過啟用採用 Amazon S3 受管金鑰 (SSE-S3) 的伺服器端加密或採用 KMS 金鑰 (SSE-KMS) 的伺服器端加密，您可以保護 Amazon S3 儲存貯體中的資料。如需詳細資訊，請參閱《Amazon S3 使用者指南》中的使用伺服器端加密保護資料。

如果您將稽核調查結果發送至以 SSE-S3 保護的儲存貯體，則不需要其他組態。Amazon S3 會處理加密金鑰。

如果您將稽核調查結果發送至以 SSE-KMS 保護的儲存貯體，您必須更新 KMS 金鑰的金鑰政策，讓日誌傳遞帳戶能夠寫入您的 S3 儲存貯體。如需使用 SSE-KMS 所需之金鑰政策的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 Amazon S3。

您可以保護的資料類型

本節包含有關您可以在 CloudWatch Logs 資料保護政策中保護的資料類型的資訊。CloudWatch Logs 受管資料識別符提供預先設定的資料類型，以保護財務資料、個人健康資訊 (PHI) 和個人身分識別資訊 (PII)。您也可以使用自訂資料識別符來建立針對特定使用案例量身打造的資料識別符。

內容

適用於敏感資料類型的 CloudWatch Logs 受管資料識別符

本節包含您可以使用受管資料識別符保護的資料類型，以及哪些國家和區域與每種資料類型相關的相關資訊。

對於某些類型的敏感資料，CloudWatch Logs 資料保護功能會掃描資料相鄰的關鍵字，並僅在找到該關鍵字時確定尋找到相符項目。如果關鍵字必須接近特定類型的資料，則關鍵字通常必須在資料的 30 個字元 （ 包含 ） 內。

如果關鍵字包含空格，CloudWatch Logs 資料保護功能會自動比對缺少空格，或包含底線 (_) 或連字號 (-) (而非空格) 的關鍵字變化。在某些情況下，CloudWatch Logs 還會擴展或縮寫關鍵字，以因應關鍵字的常見變化。

下表列出 CloudWatch Logs 可以使用受管資料識別符偵測的憑證、裝置、財務、醫療和受保護醫療資訊 (PHI) 類型。這些是某些資料類型的個人身分識別資訊 (PII) 等資料。

支援的識別符 (與語言和區域無關)

識別符	Category
`Address`	個人
`AwsSecretKey`	憑證
`CreditCardExpiration`	金融
`CreditCardNumber`	金融
`CreditCardSecurityCode`	金融

識別符	Category
EmailAddress	個人
IpAddress	個人
LatLong	個人
Name	個人
OpenSshPrivateKey	憑證
PgpPrivateKey	憑證
PkcsPrivateKey	憑證
PuttyPrivateKey	憑證
VehicleIdentificationNumber	個人

與區域相關的資料識別符需要包含識別符名稱、一個連字號，以及兩個字母 (ISO 3166-1 alpha-2) 代碼。例如 DriversLicense-US。

支援的識別符 (必須包含兩個字母的國家或地區碼)

識別符	Category	國家/地區與語言
BankAccountNumber	金融	DE、ES、FR、GB、IT、US
CepCode	個人	BR
Cnpj	個人	BR
CpfCode	個人	BR
DriversLicense	個人	AT、AU、BE、 BG、CA、CY、 CZ、DE、DK、EE、ES、FI、 FR、GB、GR、 HR、HU、IE、IT、LT、LU、

識別符	Category	國家/地區與語言
		LV、MT、NL、PL、PT、RO、SE、SI、SK、US
DrugEnforcementAgencyNumber	運作狀態	US
ElectoralRollNumber	個人	GB
HealthInsuranceCardNumber	運作狀態	歐盟
HealthInsuranceClaimNumber	運作狀態	US
HealthInsuranceNumber	運作狀態	法國
HealthcareProcedureCode	運作狀態	US
IndividualTaxIdentificationNumber	個人	美國
InseeCode	個人	法國
MedicareBeneficiaryNumber	運作狀態	US
NationalDrugCode	運作狀態	US
NationalIdentificationNumber	個人	DE、ES、IT
NationalInsuranceNumber	個人	GB
NationalProviderId	運作狀態	US
NhsNumber	運作狀態	GB
Nienumber	個人	ES
NifNumber	個人	ES
PassportNumber	個人	CA、DE、ES、FR、GB、IT、US

識別符	Category	國家/地區與語言
PermanentResidenceNumber	個人	CA
PersonalHealthNumber	醫療保健	CA
PhoneNumber	個人	BR、DE、ES、FR、GB、IT、US
PostalCode	個人	CA
RgNumber	個人	BR
SocialInsuranceNumber	個人	CA
Ssn	個人	ES、US
TaxId	個人	DE、ES、FR、GB
ZipCode	個人	美國

憑證

CloudWatch Logs 資料保護功能可以找到下列類型的憑證。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
AWS 私密存取金鑰	`AwsSecretKey`	`aws_secret_access_key` , `credentials` , `secret access key`, `secret key`, `set-awscredential`	全部
OpenSSH 私密金鑰	`OpenSSHPrivateKey`	無	全部
PGP 私密金鑰	`PgpPrivateKey`	無	全部

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
Pkcs 私有金鑰	PkcsPrivateKey	無	全部
PuTTY 私密金鑰	PuttyPrivateKey	無	全部

憑證資料類型的資料識別符 ARN

以下列出您可新增至資料保護政策的資料識別符 Amazon Resource Name (ARN)。

憑證資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/AwsSecretKey
arn:aws:dataprotection::aws:data-identifier/OpenSshPrivateKey
arn:aws:dataprotection::aws:data-identifier/PgpPrivateKey
arn:aws:dataprotection::aws:data-identifier/PkcsPrivateKey
arn:aws:dataprotection::aws:data-identifier/PuttyPrivateKey

裝置識別符

CloudWatch Logs 資料保護功能可以找到下列類型的裝置識別符。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
IP 位址	IpAddress	無	全部

裝置資料類型的資料識別符 ARN

以下列出您可新增至資料保護政策的資料識別符 Amazon Resource Name (ARN)。

裝置資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/IpAddress

財務資訊

CloudWatch Logs 資料保護功能可以找到下列類型的財務資訊。

如果您設定了資料保護政策，則無論日誌群組所在的地理位置為何，CloudWatch Logs 都會掃描您指定的資料識別符。此資料表中國家與地區資料欄中的資訊指出，是否必須在資料識別符後附加兩個字母的國家/地區碼，以偵測這些國家和地區的相應關鍵字。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
銀行帳戶號碼	BankAccountNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的銀行帳戶號碼的關鍵字資料表。	法國、德國、義大利、西班牙、英國、美國	包含最多由 34 個英數字元組成的國際銀行帳號 (IBANs)，包括國家/地區代碼等元素。
信用卡到期日	CreditCardExpiration	exp d, exp m, exp y, expiration , expiry	全部	
信用卡號碼	CreditCardNumber	account number, american express, amex, bank card, card,	全部	偵測要求資料是符合

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
		card number, card num, cc #, ccn, check card, credit, credit card#, dankort, debit, debit card, diners club, discover, electron, japanese card bureau, jcb, mastercard , mc, pan, payment account number, payment card number, pcn, union pay, visa		Luhn 檢查公式的 13-19 位數序列，並針對以下任何類型的信用卡使用標準卡號字首：American Express、Dankort、Diner's Club、Discover、Electron、Japanese Card Bureau (JCB)、Mastercard、UnionPay 和 Visa。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
信用卡驗證碼	CreditCardSecurityCode	card id, card identification code, card identification number , card security code, card validation code , card validation number , card verification data , card verification value, cvc, cvc2, cvv, cvv2, elo verification code	全部	

銀行帳戶號碼的關鍵字

使用下列關鍵字處理銀行帳戶號碼。這包括國際銀行帳號 (IBANs)，最多可包含 34 個英數字元，包括國家/地區代碼等元素。

Country	關鍵字
法國	account code, account number, accountno# , accountnumber# , bban, code bancaire, compte bancaire, customer account id, customer account number, customer bank account id, iban, numéro de compte
德國	account code, account number, accountno# , accountnumber# , bankleitzahl , bban, customer account id, customer account number, customer bank account id, geheimzahl , iban, kartennummer , kontonummer , kreditkartennummer , sepa
義大利	account code, account number, accountno# , accountnumber# , bban, codice bancario, conto bancario, customer account id,

Country	關鍵字
	customer account number, customer bank account id, iban, numero di conto
西班牙	account code, account number, accountno# , accountnumber# , bban, código cuenta, código cuenta bancaria, cuenta cliente id, customer account ID, customer account number, customer bank account id, iban, número cuenta bancaria cliente, número cuenta cliente
英國	account code, account number, accountno# , accountnumber# , bban, customer account ID, customer account number, customer bank account id, iban, sepa
美國	bank account, bank acct, checking account, checking acct, deposit account, deposit acct, savings account, savings acct, chequing account, chequing acct

CloudWatch Logs 不會報告出現的以下序列，信用卡發卡機構已保留這些序列供公開測試使用。

```
122000000000003, 2222405343248877, 2222990905257051, 2223007648726984,
 2223577120017656,
30569309025904, 34343434343434, 3528000700000000, 3530111333300000, 3566002020360505,
 36148900647913,
36700102000000, 371449635398431, 378282246310005, 378734493671000, 38520000023237,
 4012888888881881,
4111111111111111, 4222222222222, 4444333322221111, 4462030000000000, 4484070000000000,
 4911830000000,
4917300800000000, 4917610000000000, 4917610000000000003, 5019717010103742,
 5105105105105100,
5111010030175156, 5185540810000019, 5200828282828210, 5204230080000017,
 5204740009900014, 5420923878724339,
5454545454545454, 5455330760000018, 5506900490000436, 5506900490000444,
 5506900510000234, 5506920809243667,
5506922400634930, 5506927427317625, 5553042241984105, 5555553753048194,
 5555555555554444, 5610591081018250,
6011000990139424, 6011000400000000, 6011111111111117, 630490017740292441,
 630495060000000000,
6331101999990016, 6759649826438453, 6799990100000000019, and 76009244561.
```

財務資料類型的資料識別符 ARN

以下列出您可新增至資料保護政策的資料識別符 Amazon Resource Name (ARN)。

財務資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-DE
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-ES
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-FR
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-GB
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-IT
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-US
arn:aws:dataprotection::aws:data-identifier/CreditCardExpiration
arn:aws:dataprotection::aws:data-identifier/CreditCardNumber
arn:aws:dataprotection::aws:data-identifier/CreditCardSecurityCode

受保護醫療資訊 (PHI)

CloudWatch Logs 資料保護功能可以找到下列類型的受保護醫療資訊 (PHI)。

如果您設定了資料保護政策，則無論日誌群組所在的地理位置為何，CloudWatch Logs 都會掃描您指定的資料識別符。此資料表中國家與地區資料欄中的資訊指出，是否必須在資料識別符後附加兩個字母的國家/地區碼，以偵測這些國家和地區的相應關鍵字。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
緝毒署 (DEA) 註冊號碼	DrugEnforcementAgencyNumber	dea number, dea registration	美國

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
健康保險卡號碼 (EHIC)	HealthInsuranceCardNumber	assicurazione sanitaria numero, carta assicurazione numero, carte d’assurance maladie , carte européenne d'assurance maladie , ceam, ehic, ehic#, finlandehicnumber# , gesundheitskarte , hälsokort , health card, health card number, health insurance card, health insurance number, insurance card number, krankenversicherungskarte , krankenversicherungsnummer , medical account number, numero conto medico, numéro d’assurance maladie , numéro de carte d’assurance , numéro de compte medical, número de cuenta médica, número de seguro de salud, número de tarjeta de seguro, sairaanhoitokortin , sairausvakuutuskortti , sairausvakuutusnumero , sjukförsäkring	歐盟

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
		nummer, sjukförsäkringskort , suomi ehic-numero , tarjeta de salud, terveyskortti , tessera sanitaria assicurazione numero , versicherungsnummer	
健康保險索償編碼 (HICN)	HealthInsuranceClaimNumber	health insurance claim number, hic no, hic no., hic number, hic#, hicn, hicn#, hicno#	美國
健康保險或醫療識別號碼	HealthInsuranceNumber	carte d'assuré social, carte vitale, insurance card	法國
醫療保健通用程序編碼系統 (HCPCS) 代碼	HealthcareProcedureCode	current procedural terminology , hcpcs, healthcare common procedure coding system	美國
聯邦醫療保險受益人號碼 (MBN)	MedicareBeneficiaryNumber	mbi, medicare beneficiary	美國
國家藥物法規 (NDC)	NationalDrugCode	national drug code, ndc	美國
國家提供者識別符 (NPI)	NationalProviderId	hipaa, n.p.i., national provider, npi	美國

資料類型	資料識別符 ID	必要的關鍵字	國家和地區
國民保健署 (NHS) 號碼	NhsNumber	national health service, NHS	英國
個人健康號碼	PersonalHealthNumber	canada healthcare number, msp number, care number, phn, soins de santé	加拿大

受保護醫療資訊 (PHI) 資料類型的資料識別符 ARN

以下列出可用於受保護醫療資訊 (PHI) 資料保護政策的資料識別符 Amazon Resource Name (ARN)。

PHI 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/DrugEnforcementAgencyNumber-US
arn:aws:dataprotection::aws:data-identifier/HealthcareProcedureCode-US
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceCardNumber-EU
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceClaimNumber-US
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceNumber-FR
arn:aws:dataprotection::aws:data-identifier/MedicareBeneficiaryNumber-US
arn:aws:dataprotection::aws:data-identifier/NationalDrugCode-US

PHI 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/NationalInsuranceNumber-GB
arn:aws:dataprotection::aws:data-identifier/NationalProviderId-US
arn:aws:dataprotection::aws:data-identifier/NhsNumber-GB
arn:aws:dataprotection::aws:data-identifier/PersonalHealthNumber-CA

個人身分識別資訊 (PII)

CloudWatch Logs 資料保護功能可以找到下列類型的個人身分識別資訊 (PII)。

如果您設定了資料保護政策，則無論日誌群組所在的地理位置為何，CloudWatch Logs 都會掃描您指定的資料識別符。此資料表中國家與地區資料欄中的資訊指出，是否必須在資料識別符後附加兩個字母的國家/地區碼，以偵測這些國家和地區的相應關鍵字。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
出生日期	DateOfBirth	dob, date of birth, birthdate , birth date, birthday, b-day, bday	任何	支援大多數日期格式，例如所有數字以及數字和月份名稱的組合。您可以用空格、斜線 (/) 或連字號

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
				(-) 分隔日期組成部分。
Código de Endereçamento Postal (CEP)	CepCode	cep, código de endereçamento postal, codigo de endereçamento postal	巴西	
Cadastro Nacional da Pessoa Jurídica (CNPJ)	Cnpj	cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj	巴西	
Cadastro de Pessoas Físicas (CPF)	CpfCode	Cadastro de pessoas fisicas, cadastro de pessoas físicas, cadastro de pessoa física, cadastro de pessoa fisica, cpf	巴西	
駕照識別號碼	DriversLicense	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的駕照識別號碼資料表。	許多國家/地區。如需詳細資訊，請參閱駕照識別號碼資料表。	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
選民名冊號碼	ElectoralRollNumber	electoral #, electoral number, electoral roll #, electoral roll no., electoral roll number, electoral rollno	英國	
個人納稅識別號碼	IndividualTaxIdentificationNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的個人納稅人識別號碼資料表。	巴西、法國、德國、西班牙、英國	
國家統計和經濟研究所 (INSEE)	InseeCode	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的國民身分證號碼的關鍵字資料表。	法國	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
國民身分證號碼	NationalIdentificationNumber	是。如需詳細資訊，請參閱本節後文的國民身分證號碼的關鍵字資料表。	德國、義大利、西班牙	這包括 Documento Nacional de Identidad (DNI) 識別符 (西班牙)、Codice fiscale codes (義大利) 和國民身分證號碼 (德國)。
國民保險號碼 (NINO)	NationalInsuranceNumber	insurance no., insurance number, insurance# , national insurance number, nationalinsurance# , nationalinsurancenumber , nin, nino	英國	–
Número de identidad de extranjero (NIE)	NieNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的個人納稅人識別號碼資料表。	西班牙	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
Número de Identificación Fiscal (NIF)	NifNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的個人納稅人識別號碼資料表。	西班牙	
護照號碼	PassportNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的護照號碼的關鍵字資料表。	加拿大、法國、德國、義大利、西班牙、英國、美國	
永久居留號碼	Permanent Residence Number	carte résident permanent , numéro carte résident permanent , numéro résident permanent , permanent resident card, permanent resident card number, permanent resident no, permanent resident no., permanent resident number, pr no, pr no., pr non, pr number, résident permanent no., résident permanent non	加拿大	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
電話號碼	PhoneNumber	巴西：關鍵字還包括：cel、celular、fone、m residencial 、numero residencial 、telefone 其他：cell、contact、fax、number、mobile、phone、number、tel、telephone 、telephone number	巴西、加拿大、法國、德國、義大利、西班牙、英國、美國	這包括美國免付費電話號碼和傳真號碼。如果關鍵字與資料相鄰，則該號碼不必包含國家/地區代碼。如果關鍵字不在資料附近，則該數字必須包含國家/地區代碼。
郵遞區號	PostalCode	無	加拿大	
Registro Geral (RG)	RgNumber	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的個人納稅人識別號碼資料表。	巴西	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
社會保險號碼 (SIN)	SocialInsuranceNumber	canadian id, numéro d'assurance sociale, social insurance number, sin	加拿大	
社會安全號碼 (SSN)	Ssn	西班牙 – número de la seguridad social、social security no.、social security no、número de la seguridad social、social security number、socialsecurityno# 、ssn、ssn# 美國 – social security、ss#、ssn	西班牙、美國	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
納稅識別號碼或參考號碼	TaxId	是。不同的關鍵字適用於不同的國家/地區。如需詳細資訊，請參閱本節後文的個人納稅人識別號碼資料表。.	法國、德國、西班牙、英國	這包括 TIN (法國)；Steueridentifikationsnummer (德國)；CIF (西班牙)；以及 TRN、UTR (英國)。
郵遞區號	ZipCode	zip code, zip+4	美國	美國郵遞區號。
郵寄地址	Address	無	澳洲、加拿大、法國、德國、義大利、西班牙、英國、美國	雖然不需要使用關鍵字，但偵測需要地址中包含城市或地點的名稱以及郵遞區號。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
電子郵件地址	EmailAddress	無	任何	

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
全球定位系統 (GPS) 座標	LatLong	coordinate ，coordinates ，lat long， latitude longitude ，location，position	任何	如果緯度和經度座標以一對形式儲存，且使用十進位度 (DD) 格式 (例如 41.948614,-87.655311)，CloudWatch Logs 就可以偵測 GPS 座標。支援不包括度數十進位分鐘 (DDM) 格式的座標 (例如 41°56.9168'N 87°39.3187'W) 或度、

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
				分、秒 (DMS) 格式 (例如 41°56'55.0104"N 87°39'19.1196"W)。
全名	Name	無	任何	CloudWatch Logs 只能偵測全名。支援僅限於拉丁字元集。

資料類型	資料識別符 ID	必要的關鍵字	國家和地區	備註
車輛識別符 (VIN)	VehicleIdentificationNumber	Fahrgestellnummer , niv, numarul de identificare , numarul seriei de sasiu, serie sasiu, numer VIN, Número de Identificação do Veículo, Número de Identificación de Automóviles , numéro d'identification du véhicule, vehicle identification number, vin, VIN numeris	任何	CloudWatch Logs 可以偵測包含 17 個字元序列並符合 ISO 3779 和 3780 標準的 VIN。這些標準是專為全球使用而設計的。

駕照識別號碼的關鍵字

為了偵測各種類型的駕照識別號碼，CloudWatch Logs 要求關鍵字與這些號碼相鄰。下表列出 CloudWatch Logs 可辨識的特定國家和地區的關鍵字。

國家/地區或區域	關鍵字
澳洲	dl# dl:, dl :, dlno# driver licence, driver license, driver permit, drivers lic., drivers licence, driver's licence, drivers license, driver's license, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit

國家/地區或區域	關鍵字
奧地利	führerschein, fuhrerschein, führerschein republik österreich, fuhrerschein republik osterreich
比利時	fuehrerschein, fuehrerschein- nr, fuehrersc heinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrersch einnummer, führerscheinnummer, numéro permis conduire, permis de conduire, rijbewijs, rijbewijsnummer
保加利亞	превозно средство, свидетелство за управление на моторно, свидетелство за управление на мпс, сумпс, шофьорска книжка
加拿大	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit, permis de conduire
克羅埃西亞	vozačka dozvola
賽普勒斯	άδεια οδήγησης
捷克	číslo licence, císlo licence řidiče, číslo řidičskéh o průkazu, ovladače lic., povolení k jízdě, povolení řidiče, řidiči povolení, řidičský prúkaz, řidičský průkaz
丹麥	kørekort, kørekortnummer

國家/地區或區域	關鍵字
愛沙尼亞	juhi litsentsi number, juhiloa number, juhiluba, juhiluba number
芬蘭	ajokortin numero, ajokortti, förare lic., körkort, körkort nummer, kuljettaja lic., permis de conduire
法國	permis de conduire
德國	fuehrerschein, fuehrerschein- nr, fuehrersc heinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrersch einnummer, führerscheinnummer
希臘	δεια οδήγησης, adeia odigisis
匈牙利	illesztőprogramok lic, jogosítvány, jogsi, licencszám, vezető engedély, vezetői engedély
愛爾蘭	ceadúnas tiomána
義大利	patente di guida, patente di guida numero, patente guida, patente guida numero
拉脫維亞	autovadītāja apliecība, licences numurs, vadītāja apliecība, vadītāja apliecības numurs, vadītāja atļauja, vadītāja licences numurs, vadītāji lic.
立陶宛	vairuotojo pažymėjimas
盧森堡	fahrerlaubnis, führerschäin
馬爾他	liċenzja tas-sewqan
荷蘭	permis de conduire, rijbewijs, rijbewijsnummer

國家/地區或區域	關鍵字
波蘭	numer licencyjny, prawo jazdy, zezwolenie na prowadzenie
葡萄牙	carta de condução, carteira de habilitação, carteira de motorist, carteira habilitação, carteira motorist, licença condução, licença de condução, número de licença, número licença, permissão condução, permissão de condução
羅馬尼亞	numărul permisului de conducere, permis de conducere
斯洛伐克	číslo licencie, číslo vodičského preukazu, ovládače lic., povolenia vodičov, povolenie jazdu, povolenie na jazdu, povolenie vodiča, vodičský preukaz
斯洛維尼亞	vozniško dovoljenje
西班牙	carnet conducer, el carnet de conducer, licencia conducer, licencia de manejo, número carnet conducer, número de carnet de conducer, número de permiso conducer, número de permiso de conducer, número licencia conducer, número permiso conducer, permiso conducción, permiso conducer, permiso de conducción
瑞典	ajokortin numero, dlno# ajokortti, drivere lic., förare lic., körkort, körkort nummer, körkortsn ummer, kuljettajat lic.

國家/地區或區域	關鍵字
英國	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit
美國	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit

國民身分證號碼的關鍵字

若要偵測各種類型的國民身分證號碼，CloudWatch Logs 要求關鍵字與號碼相鄰。這包括 Documento Nacional de Identidad (DNI) 識別符 (西班牙)、法國國家統計和經濟研究所 (INSEE) 代碼、德國國民身分證號碼和 Registro Geral (RG) 號碼 (巴西)。

下表列出 CloudWatch Logs 可辨識的特定國家和地區的關鍵字。

國家/地區或區域	關鍵字
巴西	registro geral, rg
法國	assurance sociale, carte nationale d’identit é, cni, code sécurité sociale, French social security number, fssn#, insee, insurance number, national id number, nationalid#, numéro d'assurance, sécurité sociale, sécurité

國家/地區或區域	關鍵字
	sociale non., sécurité sociale numéro, social, social security, social security number, socialsecuritynumber, ss#, ssn, ssn#
德國	ausweisnummer, id number, identification number, identity number, insurance number, personal id, personalausweis
義大利	codice fiscal, dati anagrafici, ehic, health card, health insurance card, p. iva, partita i.v.a., personal data, tax code, tessera sanitaria
西班牙	dni, dni#, dninúmero#, documento nacional de identidad, identidad único, identidadúnico#, insurance number, national identification number, national identity, nationalid#, nationali dno#, número nacional identidad, personal identification number, personal identity no, unique identity number, uniqueid#

護照號碼的關鍵字

若要偵測各種類型的護照號碼，CloudWatch Logs 要求關鍵字與號碼相鄰。下表列出 CloudWatch Logs 可辨識的特定國家和地區的關鍵字。

國家/地區或區域	關鍵字
加拿大	passeport, passeport#, passport, passport#, passportno, passportno#
法國	numéro de passeport, passeport, passeport #, passeport #, passeportn °, passeport n °, passeportNon, passeport non

國家/地區或區域	關鍵字
德國	ausstellungsdatum, ausstellungsort, geburtsdatum, passport, passports, reisepass, reisepass–nr, reisepassnummer
義大利	italian passport number, numéro passeport , numéro passeport italien, passaporto, passaporto italiana, passaporto numero, passport number, repubblica italiana passaporto
西班牙	españa pasaporte, libreta pasaporte, número pasaporte, pasaporte, passport, passport book, passport no, passport number, spain passport
英國	passeport #, passeport n °, passeportNon, passeport non, passeportn °, passport #, passport no, passport number, passport#, passportid
美國	passport, travel document

納稅識別號碼及參考號碼的關鍵字

為了偵測各種類型的納稅人識別和參考號碼，CloudWatch Logs 要求關鍵字與這些號碼相鄰。下表列出 CloudWatch Logs 可辨識的特定國家和地區的關鍵字。

國家/地區或區域	關鍵字
巴西	cadastro de pessoa física, cadastro de pessoa fisica, cadastro de pessoas físicas, cadastro de pessoas fisicas, cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj, cpf
法國	numéro d'identification fiscale, tax id, tax identification number, tax number, tin, tin#

國家/地區或區域	關鍵字
德國	identifikationsnummer, steuer id, steueride ntifikationsnummer, steuernummer, tax id, tax identification number, tax number
西班牙	cif, cif número, cifnúmero#, nie, nif, número de contribuyente, número de identidad de extranjero, número de identificación fiscal, número de impuesto corporativo, personal tax number, tax id, tax identification number, tax number, tin, tin#
英國	paye, tax id, tax id no., tax id number, tax identification, tax identification#, tax no., tax number, tax reference, tax#, taxid#, temporary reference number, tin, trn, unique tax reference, unique taxpayer reference, utr
美國	個人納稅人識別號碼，itin，i.t.i.n。

個人身分識別資訊 (PII) 的資料識別符 ARN

下表列出您可新增至資料保護政策的個人身分識別資訊 (PII) 資料識別符的 Amazon Resource Name (ARN)。

PII 資料識別符 ARN
`arn:aws:dataprotection::aws:data-identifier/Address`
`arn:aws:dataprotection::aws:data-identifier/CepCode-BR`
`arn:aws:dataprotection::aws:data-identifier/Cnpj-BR`
`arn:aws:dataprotection::aws:data-identifier/CpfCode-BR`
`arn:aws:dataprotection::aws:data-identifier/DriversLicense-AT`

PII 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/DriversLicense-AU
arn:aws:dataprotection::aws:data-identifier/DriversLicense-BE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-BG
arn:aws:dataprotection::aws:data-identifier/DriversLicense-CA
arn:aws:dataprotection::aws:data-identifier/DriversLicense-CY
arn:aws:dataprotection::aws:data-identifier/DriversLicense-CZ
arn:aws:dataprotection::aws:data-identifier/DriversLicense-DE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-DK
arn:aws:dataprotection::aws:data-identifier/DriversLicense-EE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-ES
arn:aws:dataprotection::aws:data-identifier/DriversLicense-FI
arn:aws:dataprotection::aws:data-identifier/DriversLicense-FR
arn:aws:dataprotection::aws:data-identifier/DriversLicense-GB
arn:aws:dataprotection::aws:data-identifier/DriversLicense-GR
arn:aws:dataprotection::aws:data-identifier/DriversLicense-HR
arn:aws:dataprotection::aws:data-identifier/DriversLicense-HU
arn:aws:dataprotection::aws:data-identifier/DriversLicense-IE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-IT
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LT
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LU

PII 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LV
arn:aws:dataprotection::aws:data-identifier/DriversLicense-MT
arn:aws:dataprotection::aws:data-identifier/DriversLicense-NL
arn:aws:dataprotection::aws:data-identifier/DriversLicense-PL
arn:aws:dataprotection::aws:data-identifier/DriversLicense-PT
arn:aws:dataprotection::aws:data-identifier/DriversLicense-RO
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SI
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SK
arn:aws:dataprotection::aws:data-identifier/DriversLicense-US
arn:aws:dataprotection::aws:data-identifier/ElectoralRollNumber-GB
arn:aws:dataprotection::aws:data-identifier/EmailAddress
arn:aws:dataprotection::aws:data-identifier/IndividualTaxIdentificationNumber-US
arn:aws:dataprotection::aws:data-identifier/InseeCode-FR
arn:aws:dataprotection::aws:data-identifier/LatLong
arn:aws:dataprotection::aws:data-identifier/Name
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-DE
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-ES

PII 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-IT
arn:aws:dataprotection::aws:data-identifier/NieNumber-ES
arn:aws:dataprotection::aws:data-identifier/NifNumber-ES
arn:aws:dataprotection::aws:data-identifier/PassportNumber-CA
arn:aws:dataprotection::aws:data-identifier/PassportNumber-DE
arn:aws:dataprotection::aws:data-identifier/PassportNumber-ES
arn:aws:dataprotection::aws:data-identifier/PassportNumber-FR
arn:aws:dataprotection::aws:data-identifier/PassportNumber-GB
arn:aws:dataprotection::aws:data-identifier/PassportNumber-IT
arn:aws:dataprotection::aws:data-identifier/PassportNumber-US
arn:aws:dataprotection::aws:data-identifier/PermanentResidenceNumber-CA
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-BR
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-DE
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-ES
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-FR
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-GB
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-IT
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-US
arn:aws:dataprotection::aws:data-identifier/PostalCode-CA

PII 資料識別符 ARN
arn:aws:dataprotection::aws:data-identifier/RgNumber-BR
arn:aws:dataprotection::aws:data-identifier/SocialInsuranceNumber-CA
arn:aws:dataprotection::aws:data-identifier/Ssn-ES
arn:aws:dataprotection::aws:data-identifier/Ssn-US
arn:aws:dataprotection::aws:data-identifier/TaxId-DE
arn:aws:dataprotection::aws:data-identifier/TaxId-ES
arn:aws:dataprotection::aws:data-identifier/TaxId-FR
arn:aws:dataprotection::aws:data-identifier/TaxId-GB
arn:aws:dataprotection::aws:data-identifier/VehicleIdentificationNumber
arn:aws:dataprotection::aws:data-identifier/ZipCode-US

自訂資料識別符

主題

什麼是 SNS 自訂資料識別符？

自訂資料識別符 (CDI) 可讓您定義自訂的規則運算式，以用於資料保護政策。使用自訂資料識別符就可以鎖定受管資料識別符無法提供的企業特定個人身分識別資訊 (PII) 使用案例。例如，您可以使用自訂資料識別符來尋找公司專屬員工 ID。自訂資料識別符可與受管資料識別符搭配使用。

自訂資料識別符的限制

CloudWatch Logs 自訂資料識別符有下列限制：

- 每個資料保護政策最多可支援 10 個自訂資料識別符。
- 自訂資料識別符名稱的長度上限為 128 個字元。支援的字元如下：
  - 英數字：(a-zA-Z0-9)
  - 符號：('_' | '-')
- RegEx 的長度上限為 200 個字元。支援的字元如下：
  - 英數字：(a-zA-Z0-9)
  - 符號：('_' | '#' | '=' | '@' |'/' | ';' | ',' | '-' | ' ')
  - RegEx 保留字元：('^' | '$' | '?' | '[' | ']' | '{' | '}' | '|' | '\\' | '*' | '+' | '.')
- 自訂資料識別符的名稱不可與受管資料識別符的名稱相同。
- 您可以在帳戶層級資料保護政策或日誌群組層級資料保護政策中指定自訂資料識別符。與受管資料識別符類似，帳戶層級政策中定義的自訂資料識別符可與日誌群組層級政策中定義的自訂資料識別符搭配使用。

在主控台中使用自訂資料識別符

當您使用 CloudWatch 主控台建立或編輯資料保護政策時，若要指定自訂資料識別符，您只需輸入資料識別符的名稱和規則表達式。例如，您可以**Employee_ID**輸入 **EmployeeID-\d{9}**做為名稱和規則表達式。此規則表達式會在 之後偵測和遮罩具有九個數字的日誌事件EmployeeID-。例如 EmployeeID-123456789

在您的資料保護政策中使用自訂資料識別符

如果您使用 AWS CLI 或 AWS API 來指定自訂資料識別符，則需要在用來定義資料保護政策的 JSON 政策中包含資料識別符名稱和規則表達式。下列資料保護政策會偵測和遮罩帶有公司特定員工 IDs日誌事件。

1. 在您的資料保護政策內建立 Configuration 區塊。
2. 輸入自訂資料識別符的 Name。例如 **EmployeeId**。
3. 輸入自訂資料識別符的 Regex。例如 **EmployeeID-\d{9}**。此規則表達式將符合包含 之後有九位數EmployeeID-的日誌事件EmployeeID-。例如 EmployeeID-123456789
4. 請參閱政策聲明中的下列自訂資料識別符。

```
{
    "Name": "example_data_protection_policy",
    "Description": "Example data protection policy with custom data identifiers",
    "Version": "2021-06-01",
    "Configuration": {
      "CustomDataIdentifier": [
        {"Name": "EmployeeId", "Regex": "EmployeeId-\\d{9}"}
      ]
    },
    "Statement": [
        {
            "Sid": "audit-policy",
            "DataIdentifier": [
                "EmployeeId"
            ],
            "Operation": {
                "Audit": {
                    "FindingsDestination": {
                        "S3": {
                            "Bucket": "EXISTING_BUCKET"
                        }
                    }
                }
            }
        },
        {
            "Sid": "redact-policy",
            "DataIdentifier": [
            "EmployeeId"
            ],
            "Operation": {
                "Deidentify": {
                    "MaskConfig": {
                    }
                }
            }
        }
    ]
}
```

5. (選用) 視需要繼續將其他自訂資料識別符新增至 Configuration 區塊。資料保護政策目前最多可支援 10 個自訂資料識別符。

在擷取期間轉換日誌

透過日誌轉換和擴充，您可以在擷取至 CloudWatch Logs 時，以一致且內容豐富的格式將所有日誌標準化。您可以使用 AWS WAF 和 Amazon Route 53 等常見 AWS 服務的預先設定範本，或使用 Grok 等原生剖析器建置自訂轉換器，將結構新增至日誌。您也可以重新命名現有的屬性，並將其他中繼資料新增至您的日誌，例如帳戶 ID 和區域。

日誌轉換有助於簡化和縮短跨應用程式的日誌查詢，並有助於簡化在日誌上建立提醒。此功能使用 VPC Flow 日誌、Route 53 和 等主要日誌來源out-of-the-box轉換範本，為常見的 AWS 日誌類型提供轉換 Amazon RDS for PostgreSQL。您可以使用預先設定的轉換範本，或建立自訂轉換器以符合您的需求。

日誌轉換可協助您管理從各種來源發出的日誌，這些來源的格式和屬性名稱差異很大。

建立轉換器之後，擷取的日誌事件會以標準格式轉換和儲存。您可以使用下列功能，利用這些轉換的日誌來加速分析體驗：

- 欄位索引
- CloudWatch Logs Insights 探索的欄位
- 使用指標篩選條件警示的彈性
- 透過訂閱篩選條件轉送
- 使用 Contributor Insights 從日誌事件建立指標資料，您可以選擇讓 Contributor Insights 規則在日誌事件轉換之前或之後評估日誌事件。

轉換只會在日誌擷取期間發生。您無法轉換已擷取的日誌事件。轉換無法還原。原始日誌和轉換日誌都存放在具有相同保留政策的 CloudWatch Logs 中。日誌轉換和擴充功能包含在現有的標準日誌類別擷取價格中。日誌儲存成本將根據轉換後的日誌大小而定，這可能超過原始日誌磁碟區。

> **Important**
>
> 日誌事件轉換後，您必須使用 CloudWatch Logs Insights 查詢來檢視日誌的轉換版本。GetLogEvents 和 FilterLogEvents 動作只會在轉換日誌事件之前傳回原始版本的日誌事件。

> **Important**
>
> 雖然 PutLogEvents 中允許最多 1MB 的單一日誌事件，但日誌轉換只能處理大小小於 512kb 的日誌事件。任何大於 512kb 的日誌事件都會在轉換中失敗並發出錯誤。PutLogEvents 的總大小仍然可以超過 512kb。

除了轉換為不同的格式之外，您還可以使用帳戶 ID、區域和關鍵字等其他內容來豐富您的日誌。這些是從日誌群組名稱和靜態關鍵字擷取。

日誌轉換可協助您處理從各種來源發出的日誌，這些來源的格式和屬性名稱差異很大。

只有標準日誌類別中的日誌群組才支援日誌轉換和擴充。

您可以為個別日誌群組建立轉換器，也可以建立套用至您帳戶中所有或多個日誌群組的帳戶層級轉換器。如果日誌群組具有日誌群組層級轉換器，則該轉換器會覆寫任何帳戶層級轉換器，否則會套用至該日誌群組。

主題

建立和管理日誌轉換器

日誌轉換器包含一或多個位於邏輯管道中的處理器。每個處理器都會套用到日誌事件，依序按照轉換器組態中列出的順序。

有些處理器屬於剖析器類型。每個轉換器必須至少有一個剖析器，而且轉換器中的第一個處理器必須是剖析器。

有些剖析器是內建的剖析器，這些剖析器針對特定類型的 AWS 付費日誌進行設定。

其他處理器類型包括字串變動器、JSON 變動器和資料處理器。

您可以為個別日誌群組建立轉換器，也可以建立套用至您帳戶中所有或多個日誌群組的帳戶層級轉換器。如果日誌群組具有日誌群組層級轉換器，則該轉換器會覆寫任何帳戶層級轉換器，否則會套用至該日誌群組。您帳戶中的區域最多可有 20 個帳戶層級轉換器。

建立轉換器時，必須遵循下列準則：

- 如果您包含已 AWS 佈建日誌類型的預先設定剖析器，它必須是轉換器中列出的第一個處理器。您只能在轉換器中包含一個此類處理器。
- 您只能在轉換器中包含一個`grok`處理器。
- 您必須在轉換器中至少有一個剖析器類型處理器。您可以包含最多五個剖析器類型處理器。此限制為 5 個，包括內建剖析器和可設定的剖析器。
- 您可以在轉換器中擁有多達 20 個處理器。
- 您只能在轉換器中包含一個 addKeys 處理器。
- 您只能在轉換器中包含一個 copyValue 處理器。
- 每個轉換器最多可以從日誌事件中擷取 200 個欄位。
- 每個日誌事件都必須低於 512KB。日誌事件的總大小仍然可以超過 512KB。

主題

建立帳戶層級轉換器政策

使用本節中的步驟來建立適用於帳戶中的所有日誌群組，或是具有以相同字串（字首）開頭之日誌群組名稱的多個日誌群組的轉換器政策。一個區域中最多可以有 20 個帳戶層級轉換器政策。

您無法在相同區域中建立兩個使用相同字首的轉換器政策，或另一個字首包含一個字首。例如，如果您為字串字首 建立一個轉換器政策`/aws/lambda`，則無法建立字首為 的另一個轉換器政策`/aws`。但是，您可以有一個轉換器用於 `/aws/lambda`，另一個用於 `/aws/waf`

建立帳戶層級轉換器政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇設定，然後選擇日誌索引標籤。
3. 在帳戶轉換器政策區段中，選擇建立轉換器政策。
4. 針對轉換器政策名稱，輸入新 poiicy 的名稱。
5. 對於選取日誌群組，請執行下列其中一項操作：
   - 選擇所有標準日誌群組，讓轉換器政策套用至帳戶中的所有標準類別日誌群組。
   - 依字首比對選擇日誌群組，將政策套用至名稱以相同字串開頭的日誌群組子集。然後，在選擇條件中輸入這些日誌群組的字首。
6. 在選取剖析器區域中，使用剖析器選取要包含在轉換器中的剖析器。

   如果這是已 AWS 佈建日誌類型的預先設定剖析器，則不需要為其指定任何組態。

   如果是不同的剖析器，您需要指定其組態。如需詳細資訊，請參閱 中該處理器的資訊可設定的剖析器類型處理器。
7. 若要新增另一個處理器，請選擇選取處理器。然後在處理器方塊中選取您想要的處理器，然後填入組態參數。

   請記住，處理器會以您將其新增至轉換器的順序在日誌事件上操作。
8. （選用）若要新增其他處理器，請選擇 + 處理器並重複上一個步驟。
9. （選用）您可以隨時測試您目前在範例日誌事件上建置的轉換器。若要這樣做，請在轉換器預覽區段中執行下列其中一項操作：
   - 在選取日誌群組中選取最多五個日誌群組，然後選擇載入最新的日誌事件。然後選擇測試轉換器。
   - 將日誌事件直接複製到範例日誌事件，然後選擇測試轉換器。

   接著會出現轉換後的日誌版本。
10. 當您完成新增處理器並對範例日誌的測試感到滿意時，請選擇儲存。
11. 當您完成時，請選擇 Create (建立)。

編輯或刪除帳戶層級轉換器政策

使用本節中的步驟來編輯或刪除帳戶層級轉換器政策。

編輯或刪除帳戶層級轉換器政策

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇設定，然後選擇日誌索引標籤。
3. 在轉換器帳戶政策區段中，選擇管理。
4. 依您要管理的轉換器政策選取按鈕，然後選擇編輯或刪除。

   如果您要編輯政策，請參閱 中的步驟 5-11可設定的剖析器類型處理器，查看您的選項。

從頭開始建立log-group-level日誌轉換器

使用這些步驟從頭開始建立log-group-level轉換器。

使用主控台為日誌群組建立日誌轉換器

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。
3. 選擇您要為其建立轉換器的日誌群組。
4. 選擇轉換器索引標籤。您可能需要將標籤清單向右捲動才能查看。
5. 選擇建立轉換器。
6. 在選擇剖析器方塊中，選取要包含在轉換器中的剖析器。

   如果這是已 AWS 佈建日誌類型的預先設定剖析器，則不需要為其指定任何組態。

   如果是不同的剖析器，您需要指定其組態。如需詳細資訊，請參閱 中該處理器的資訊可設定的剖析器類型處理器。
7. 若要新增另一個處理器，請選擇 + 新增處理器。然後在選擇處理器方塊中選取您想要的處理器，然後填入組態參數。

   請記住，處理器會以您將其新增至轉換器的順序在日誌事件上操作。
8. （選用）您可以隨時測試您目前在範例日誌事件上建置的轉換器。若要這樣做，請執行下列操作：

   - 在轉換預覽區段中，選擇載入範例日誌，以從此轉換器所在的日誌群組載入範例日誌事件，或將日誌事件貼到文字方塊中。

     選擇測試轉換器。日誌的轉換版本隨即出現

9. 當您完成新增處理器並對範例日誌的測試感到滿意時，請選擇儲存。

使用 從頭 AWS CLI 建立日誌轉換器

- 使用 `aws logs put-transformer` 命令。使用 `parseJSON`做為第一個處理器時，您必須使用 `@message`做為來源欄位來剖析整個日誌事件。在初始 JSON 剖析之後，您可以在後續處理器中操作特定欄位。以下是建立包含 `parseJSON`和 `addKeys` 處理器之轉換器的範例：

```
 aws logs put-transformer \
  --transformer-config '[{"parseJSON":{"source":"@message"}},{"addKeys":
{"entries":[{"key":"metadata.transformed_in","value":"CloudWatchLogs"},
{"key":"feature","value":"Transformation"}]}},{"trimString":{"withKeys":
["status"]}}]' \
  --log-group-identifier my-log-group-name
```

複製現有的log-group-level轉換器

您可以使用 主控台來複製現有轉換器的 JSON 組態。然後，您可以使用該程式碼，使用 建立相同的轉換器 AWS CLI，也可以先修改組態。

若要透過複製現有的日誌轉換器來建立日誌轉換器

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。
3. 選擇具有您要複製之轉換器的日誌群組。
4. 選擇轉換索引標籤。您可能需要將標籤清單向右捲動才能查看。
5. 選擇管理轉換器。
6. 選擇複製轉換器。這會將轉換器 JSON 複製到剪貼簿。
7. 建立 檔案並貼入轉換器組態。在此範例中，我們將呼叫 檔案 `CopiedTransformer.json`
8. 使用 AWS CLI 建立具有該組態的新轉換器。

```
aws logs put-transformer --log-group-identifier my-log-group-name \
--transformer-config file://CopiedTransformer.json
```

編輯log-group-level轉換器

使用這些步驟來編輯現有的日誌轉換器。

編輯日誌轉換器

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。
3. 選擇具有您要編輯之轉換器的日誌群組。
4. 選擇轉換索引標籤。您可能需要將標籤清單向右捲動才能查看。
5. 選擇管理轉換器。
6. 在剖析器和處理器區段中，進行變更。
7. 若要新增另一個處理器，請選擇 + 新增處理器。然後在處理器方塊中選取您想要的處理器，然後填入組態參數。

   請記住，處理器會以您將其新增至轉換器的順序在日誌事件上操作。
8. （選用）您可以隨時測試您目前在範例日誌事件上建置的轉換器。若要這樣做，請執行下列操作：

   - 在轉換預覽區段中，選擇載入範例日誌，從此轉換器的日誌群組載入範例日誌事件，或將日誌事件貼到文字方塊中。

     選擇測試轉換。日誌的轉換版本隨即出現
9. 當您完成新增處理器並對範例日誌的測試感到滿意時，請選擇儲存。

刪除log-group-level轉換器

使用這些步驟來刪除日誌轉換器。

刪除日誌轉換器

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。
3. 選擇具有您要編輯之轉換器的日誌群組。
4. 選擇轉換索引標籤。您可能需要將標籤清單向右捲動才能查看。
5. 選擇 刪除。

6. 在確認方塊中，選擇刪除政策。

可設定的剖析器類型處理器

本節包含您可以在日誌事件轉換器中使用的可設定資料剖析器處理器的相關資訊。

內容

parseJSON

parseJSON 處理器會剖析 JSON 日誌事件，並在目的地下插入擷取的 JSON 鍵值對。如果您未指定目的地，處理器會將索引鍵/值對放在根節點下方。使用 `parseJSON`做為第一個處理器時，您必須使用 `@message`做為來源欄位來剖析整個日誌事件。在初始 JSON 剖析之後，您可以在後續處理器中操作特定欄位。

原始`@message`內容不會變更，新的金鑰會新增至訊息。

欄位	Description	是否為必要？	預設	限制
source	要剖析之日誌事件中 欄位的路徑。使用點符號來存取子欄位。例如 `store.book`	否	`@message`	長度上限：128 巢狀金鑰深度上限：3
目的地	剖析 JSON 的目的地欄位	否	`Parent JSON node`	長度上限：128 巢狀金鑰深度上限：3

範例

假設擷取的日誌事件如下所示：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

然後，如果我們有此 parseJSON 處理器：

```
[
   {
        "parseJSON": {
            "destination": "new_key"
        }
   }
]
```

轉換的日誌事件如下。

```
{
    "new_key": {
        "outer_key": {
            "inner_key": "inner_value"
        }
    }
```

```
}
```

grok

使用 grok 處理器來剖析和建構使用模式比對的非結構化資料。此處理器也可以從日誌訊息中擷取欄位。

欄位	Description	是否為必要？	預設	限制	備註
source	在 上套用 Grok 比對的欄位路徑	否	`@messag`	長度上限：128 巢狀金鑰深度上限：3	
match	要比對日誌事件的 grok 模式	是		長度上限：512 grok 模式上限：20 某些 grok 模式類型具有個別用量限制。下列模式的任意組合最多可以使用五次：{URI，URIPARAM，URIPATHPARAM，SPACE，DATA，GREEDYDATA，GREEDYDATA_MULTILINE} Grok 模式不支援類型轉換。 對於常見日誌格式模式 (APACHE_ACCESS_LOG、NGINX_AC	查看所有支援的 Grok 模式

欄位	Description	是否為必要？	預設	限制	備註
				CESS_LOG、SYSLOG5424)，在常見日誌模式之後僅支援 DATA、GREEDYDATA 或 GREEDYDATA_MULTILINE 模式。	

Grok 模式的結構

這是支援的 grok 模式結構：

```
%{PATTERN_NAME:FIELD_NAME}
```

- PATTERN_NAME：指符合特定資料類型的預先定義規則運算式。僅支援預先定義的 grok 模式。不允許建立自訂模式。
- FIELD_NAME：將名稱指派給擷取的值。 FIELD_NAME是選用的，但如果您未指定此值，則會從轉換的日誌事件中捨棄擷取的資料。如果 FIELD_NAME使用虛線表示法（例如 "parent.child")，則會將其視為 JSON 路徑。
- 類型轉換：不支援明確類型轉換。使用 TypeConverter 處理器來轉換 grok 擷取之任何值的資料類型。

若要建立更複雜的相符表達式，您可以結合多個 grok 模式。最多可結合 20 個 grok 模式以符合日誌事件。例如，此模式組合`%{NUMBER:timestamp} [%{NUMBER:db} %{IP:client_ip}:%{NUMBER:client_port}] %{GREEDYDATA:data}`可用來從 Redis 慢速日誌項目擷取欄位，如下所示：

```
1629860738.123456 [0 127.0.0.1:6379] "SET" "key1" "value1"
```

Grok 範例

範例 1：使用 grok 從非結構化日誌中擷取欄位

範例日誌：

```
293750 server-01.internal-network.local OK "[Thread-000] token generated"
```

使用的轉換器：

```
[
     {
         "grok": {
             "match": "%{NUMBER:version} %{HOSTNAME:hostname} %{NOTSPACE:status}
 %{QUOTEDSTRING:logMsg}"
         }
    }
]
```

輸出：

```
{
  "version": "293750",
  "hostname": "server-01.internal-network.local",
  "status": "OK",
  "logMsg": "[Thread-000] token generated"
}
```

範例日誌：

```
23/Nov/2024:10:25:15 -0900 172.16.0.1 200
```

使用的轉換器：

```
[
    {
        "grok": {
            "match": "%{HTTPDATE:timestamp} %{IPORHOST:clientip}
 %{NUMBER:response_status}"
        }
    }
```

```
]
```

輸出：

```
{
  "timestamp": "23/Nov/2024:10:25:15 -0900",
  "clientip": "172.16.0.1",
  "response_status": "200"
}
```

範例 2：使用 grok 搭配 parseJSON 從 JSON 日誌事件擷取欄位

範例日誌：

```
{
    "timestamp": "2024-11-23T16:03:12Z",
    "level": "ERROR",
    "logMsg": "GET /page.html HTTP/1.1"
}
```

使用的轉換器：

```
[
     {
        "parseJSON": {}
    },
    {
         "grok": {
            "source": "logMsg",
             "match": "%{WORD:http_method} %{NOTSPACE:request} HTTP/
%{NUMBER:http_version}"
         }
    }
]
```

輸出：

```
{
  "timestamp": "2024-11-23T16:03:12Z",
  "level": "ERROR",
  "logMsg": "GET /page.html HTTP/1.1",
  "http_method": "GET",
```

```
  "request": "/page.html",
  "http_version": "1.1"
}
```

範例 3：FIELD_NAME 中具有虛線註釋的 Grok 模式

範例日誌：

```
192.168.1.1 GET /index.html?param=value 200 1234
```

使用的轉換器：

```
[
    {
        "grok": {
            "match": "%{IP:client.ip} %{WORD:method} %{URIPATHPARAM:request.uri}
 %{NUMBER:response.status} %{NUMBER:response.bytes}"
        }
    }
]
```

輸出：

```
{
  "client": {
    "ip": "192.168.1.1"
  },
  "method": "GET",
  "request": {
    "uri": "/index.html?param=value"
  },
  "response": {
    "status": "200",
    "bytes": "1234"
  }
}
```

支援的 grok 模式

下表列出grok處理器支援的模式。

一般 grok 模式

Grok 模式	Description	模式上限	範例
USERNAME 或 USER	符合一個或多個字元，可包含小寫字母 (a-z)、大寫字母 (A-Z)、數字 (0-9)、點 (.)、底線 (_) 或連字號 (-)	20	輸入： `user123.name-TEST` 模式：`%{USERNAME:name}` 輸出： `{"name": "user123.name-TEST"}`
INT	符合選用的加號或減號，後面接著一個或多個數字。	20	輸入： `-456` 模式：`%{INT:num}` 輸出： `{"num": "-456"}`
BASE10NUM	比對整數或浮點數與選用的符號和小數點	20	輸入： `-0.67` 模式：`%{BASE10NUM:num}` 輸出： `{"num": "-0.67"}`
BASE16NUM	將小數和十六進位數字與選用的符號 (+ 或 -) 和選用的 0x 字首配對	20	輸入： `+0xA1B2` 模式：`%{BASE16NUM:num}` 輸出： `{"num": "+0xA1B2"}`
POSINT	比對沒有前導零的完整正整數，包含一或多個數字 (1-9 後接 0-9)	20	輸入： `123` 模式：`%{POSINT:num}` 輸出： `{"num": "123"}`
NONNEGINT	符合任何整數（包含一或多個數字 0-9)，包括零和開頭為零的數字。	20	輸入： `007` 模式：`%{NONNEGINT:num}` 輸出： `{"num": "007"}`

Grok 模式	Description	模式上限	範例
WORD	比對由一或多個單字字元 (\w) 組成的整個單字，包括字母、數字和底線	20	輸入： `user_123` 模式：`%{WORD:user}` 輸出： `{"user": "user_123"}`
NOTSPACE	符合一或多個非空格字元。	5	輸入： `hello_world123` 模式：`%{NOTSPACE:msg}` 輸出： `{"msg": "hello_world123"}`
SPACE	符合零個或多個空格字元。	5	輸入： `" "` 模式：`%{SPACE:extra}` 輸出： `{"extra": " "}`
DATA	符合任何字元（新行除外）零次或多次，非貪婪。	5	輸入： `abc def ghi` 模式：`%{DATA:x} %{DATA:y}` 輸出： `{"x": "abc", "y": "def ghi"}`
GREEDYDATA	符合任何字元（新行除外）零次或多次，歡迎。	5	輸入： `abc def ghi` 模式：`%{GREEDYDATA:x} %{GREEDYDATA:y}` 輸出： `{"x": "abc def", "y": "ghi"}`

Grok 模式	Description	模式上限	範例

GREEDYDATA_MULTILINE	符合任何角色（包括換行）零次或多次，歡迎。	1	輸入:
			`abc`
			`def`
			`ghi`
			模式：`%{GREEDYDATA_MULTILINE:data}`
			輸出：`{"data": "abc\ndef\nghi"}`

QUOTEDSTRING	將引號字串（單引號或雙引號）與逸出字元相符。	20	輸入：`"Hello, world!"` 模式：`%{QUOTEDSTRING:msg}` 輸出：`{"msg": "Hello, world!"}`
UUID	符合標準 UUID 格式：8 個十六進位字元，後面接著三個 4 個十六進位字元群組，並以 12 個十六進位字元結尾，全部以連字號分隔。	20	輸入：`550e8400-e29b-41d4-a716-446655440000` 模式：`%{UUID:id}` 輸出：`{"id": "550e8400-e29b-41d4-a716-446655440000"}`

Grok 模式	Description	模式上限	範例
URN	符合 URN（統一資源名稱）語法。	20	輸入： `urn:isbn:0451450523` 模式：`%{URN:urn}` 輸出： `{"urn": "urn:isbn:0451450523"}`

AWS grok 模式

模式	Description	模式上限	範例
ARN	符合 AWS Amazon Resource Name (ARNs)`aws-cn`、擷取分割區 (`aws`、 或 `aws-us-gov`)、服務、區域、帳戶 ID，以及最多 5 個以斜線分隔的階層資源識別符。它將不符合冒號之間缺少資訊的 ARNs。	5	輸入： `arn:aws:iam:us-east-1:123456789012:user/johndoe` 模式：`%{ARN:arn}` 輸出： `{"arn": "arn:aws:iam:us-east-1:123456789012:user/johndoe"}`

網路 grok 模式

Grok 模式	Description	模式上限	範例
CISCOMAC	符合 4-4-4 十六進位格式的 MAC 地址。	20	輸入： `0123.4567.89AB` 模式：`%{CISCOMAC:MacAddress}` 輸出： `{"MacAddress": "0123.4567.89AB"}`

Grok 模式	Description	模式上限	範例
WINDOWSMAC	將十六進位格式的 MAC 地址與連字號相符	20	輸入： `01-23-45-67-89-AB` 模式：`%{WINDOWSMAC:MacAddress}` 輸出： `{"MacAddress": "01-23-45-67-89-AB"}`
COMMONMAC	比對十六進位格式的 MAC 地址與冒號。	20	輸入： `01:23:45:67:89:AB` 模式：`%{COMMONMAC:MacAddress}` 輸出： `{"MacAddress": "01:23:45:67:89:AB"}`
MAC	符合其中一個 CISCOMAC、WINDOWSMAC 或 COMMONMAC grok 模式	20	輸入： `01:23:45:67:89:AB` 模式：`%{MAC:m1}` 輸出： `{"m1":"01:23:45:67:89:AB"}`
IPV6	符合 IPv6 地址，包括壓縮表單和 IPv4-mapped IPv6 地址。	5	輸入： `2001:db8:3333:4444:5555:6666:7777:8888` 模式：`%{IPV6:ip}` 輸出： `{"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}`

Grok 模式	Description	模式上限	範例
IPV4	符合 IPv4 地址。	20	輸入： `192.168.0.1` 模式：`%{IPV4:ip}` 輸出： `{"ip": "192.168.0.1"}`
IP	符合 %{IPvIPv6} 支援的 IPv6 地址或 %{IPv4} 支援的 IPv4 地址IPv4	5	輸入： `192.168.0.1` 模式：`%{IP:ip}` 輸出： `{"ip": "192.168.0.1"}`
HOSTNAME 或 HOST	符合網域名稱，包括子網域	5	輸入： `server-01.internal-network.local` 模式：`%{HOST:host}` 輸出： `{"host": "server-01.internal-network.local"}`
IPORHOST	符合主機名稱或 IP 地址	5	輸入： `2001:db8:3333:4444:5555:6666:7777:8888` 模式：`%{IPORHOST:ip}` 輸出： `{"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}`

Grok 模式	Description	模式上限	範例
HOSTPORT	符合 %{IPORHOST} 模式支援的 IP 地址或主機名稱，後面接著冒號和連接埠號碼，在輸出中擷取連接埠做為「PORT」。	5	輸入： `192.168.1.1:8080` 模式：`%{HOSTPORT:ip}` 輸出： `{"ip":"192.168.1.1:8080","PORT":"8080"}`
URIHOST	符合 %{IPORHOST} 模式支援的 IP 地址或主機名稱，選擇性地後面接著冒號和連接埠號碼，如果有的話，擷取連接埠為「連接埠」。	5	輸入： `example.com:443 10.0.0.1` 模式：`%{URIHOST:host} %{URIHOST:ip}` 輸出： `{"host":"example.com:443","port":"443","ip":"10.0.0.1"}`

路徑grok 模式

Grok 模式	Description	模式上限	範例
UNIXPATH	符合 URL 路徑，可能包括查詢參數。	20	輸入： `/search?q=regex` 模式：`%{UNIXPATH:path}` 輸出： `{"path":"/search?q=regex"}`
WINPATH	符合 Windows 檔案路徑。	5	輸入： `C:\Users\John\Documents\file.txt` 模式：`%{WINPATH:path}` 輸出： `{"path": "C:\\Users\\John\\`

Grok 模式	Description	模式上限	範例
			Documents\\file.txt"}
PATH	符合 URL 或 Windows 檔案路徑	5	輸入：/search?q=regex 模式：%{PATH:path} 輸出：{"path":"/search?q=regex"}
TTY	比對終端機和虛擬終端機的 Unix 裝置路徑。	20	輸入：/dev/tty1 模式：%{TTY:path} 輸出：{"path":"/dev/tty1"}
URIPROTO	符合字母，選擇性後接加號 (+) 字元和其他字母或加號 (+) 字元	20	輸入：web+transformer 模式：%{URIPROTO:protocol} 輸出：{"protocol":"web+transformer"}
URIPATH	符合 URI 的路徑元件	20	輸入：/category/sub-category/product_name 模式：%{URIPATH:path} 輸出：{"path":"/category/sub-category/product_name"}

Grok 模式	Description	模式上限	範例
URIPARAM	符合 URL 查詢參數	5	輸入：`?param1=value1&param2=value2` 模式：`%{URIPARAM:url}` 輸出：`{"url":"?param1=value1&param2=value2"}`
URIPATHPARAM	選擇性地比對 URI 路徑，後面接著查詢參數	5	輸入：`/category/sub-category/product?id=12345&color=red` 模式：`%{URIPATHPARAM:path}` 輸出：`{"path":"/category/sub-category/product?id=12345&color=red"}`
URI	符合完整的 URI	5	輸入：`https://user:password@example.com/path/to/resource?param1=value1&param2=value2` 模式：`%{URI:uri}` 輸出：`{"path":"https://user:password@example.com/path/to/resource?param1=value1&param2=value2"}`

日期和時間 grok 模式

Grok 模式	Description	模式上限	範例
MONTH	將完整或縮寫的英文月份名稱配對為完整單字	20	輸入： `Jan` 模式：`%{MONTH:month}` 輸出： `{"month": "Jan"}` 輸入： `January` 模式：`%{MONTH:month}` 輸出： `{"month": "January"}`
MONTHNUM	比對從 1 到 12 的月份編號，單位數月份的選用前導零。	20	輸入： `5` 模式：`%{MONTHNUM:month}` 輸出： `{"month": "5"}` 輸入： `05` 模式：`%{MONTHNUM:month}` 輸出： `{"month": "05"}`
MONTHNUM	符合從 01 到 12 的兩位數月編號。	20	輸入： `05` 模式：`%{MONTHNUM2:month}`

Grok 模式	Description	模式上限	範例
			輸出： {"month": "05"}
週一	將月份中的某天從 1 比對到 31，加上選用的前導零。	20	輸入： 31 模式：%{MONTHDAY:monthDay} 輸出： {"monthDay":"31"}
YEAR	符合兩或四位數的年份	20	輸入： 2024 模式：%{YEAR:year} 輸出： {"year":"2024"} 輸入： 24 模式：%{YEAR:year} 輸出： {"year":"24"}
DAY	符合完整或縮寫的日名稱。	20	輸入： Tuesday 模式：%{DAY:day} 輸出： {"day":"Tuesday"}
HOUR	將 24 小時格式的小時與選用的前導零 (0)0-23 配對。	20	輸入： 22 模式：%{HOUR:hour} 輸出： {"hour":"22"}

Grok 模式	Description	模式上限	範例
MINUTE	符合分鐘數 (00-59)。	20	輸入： 59 模式：%{MINUTE:min} 輸出： {"min":"59"}
SECOND	比對代表秒數 (0)0-60 的數字，選擇性地後接小數點或冒號，以及小數分鐘的一個或多個數字	20	輸入： 3 模式：%{SECOND:second} 輸出： {"second":"3"} 輸入： 30.5 模式：%{SECOND:minSec} 輸出： {"minSec":"30.5"} 輸入： 30:5 模式：%{SECOND:minSec} 輸出： {"minSec":"30:5"}
TIME	以小時、分鐘和秒的格式 (H)H：mm：(s) 比對時間格式。秒包括閏秒 (0)0-60。	20	輸入： 09:45:32 模式：%{TIME:time} 輸出： {"time":"09:45:32"}

Grok 模式	Description	模式上限	範例
DATE_US	符合 (M)M/(d)d/(yy)y 或 (M)M-(d)d-(yy)yy 格式的日期。	20	輸入： 11/23/2024 模式：%{DATE_US:date} 輸出： {"date":"11/23/2024"} 輸入： 1-01-24 模式：%{DATE_US:date} 輸出： {"date":"1-01-24"}
DATE_EU	符合 (d)d/(M)M/(yy)yy、(d)d-(M)M-(yy)yy 或 (d)d.(M)M.(yy)yy。	20	輸入： 23/11/2024 模式：%{DATE_EU:date} 輸出： {"date":"23/11/2024"} 輸入： 1.01.24 模式：%{DATE_EU:date} 輸出： {"date":"1.01.24"}

Grok 模式	Description	模式上限	範例
ISO8601_TIMEZONE	將 UTC 位移 'Z' 或時區位移與選用冒號的格式【+-】(H)H(：)mm 比對。	20	輸入： +05:30 模式：%{ISO8601_TIMEZONE:tz} 輸出： {"tz":"+05:30"} 輸入： -530 模式：%{ISO8601_TIMEZONE:tz} 輸出： {"tz":"-530"} 輸入： Z 模式：%{ISO8601_TIMEZONE:tz} 輸出： {"tz":"Z"}
ISO8601_SECOND	比對代表秒數 (0)0-60 的數字，選擇性地後接小數點或冒號，以及小數秒的一個或多個數字	20	輸入： 60 模式：%{ISO8601_SECOND:second} 輸出： {"second":"60"}

Grok 模式	Description	模式上限	範例
TIMESTAMP_ISO8601	比對 ISO8601 日期時間格式 (yy)yy-(M)M-(d)dT(H)H：mm：((s)s)(Z|【+-】(H)H：mm) 與選用的秒和時區。	20	輸入： 2023-05-15T14:30:00+05:30 模式：%{TIMESTAMP_ISO8601:timestamp} 輸出： {"timestamp":"2023-05-15T14:30:00+05:30"} 輸入： 23-5-1T1:25+5:30 模式：%{TIMESTAMP_ISO8601:timestamp} 輸出： {"timestamp":"23-5-1T1:25+5:30"} 輸入： 23-5-1T1:25Z 模式：%{TIMESTAMP_ISO8601:timestamp} 輸出： {"timestamp":"23-5-1T1:25Z"}

Grok 模式	Description	模式上限	範例
DATE	使用 %{DATE_US} 比對美國格式的日期，或使用 %{DATE_EU} 比對歐洲格式的日期	20	輸入： 11/29/2024 模式：%{DATE:date} 輸出： {"date":"11/29/2024"} 輸入： 29.11.2024 模式：%{DATE:date} 輸出： {"date":"29.11.2024"}
DATESTAMP	符合 %{DATE} 後接 %{TIME} 模式，以空格或連字號分隔。	20	輸入： 29-11-2024 14:30:00 模式：%{DATESTAMP:dateTime} 輸出： {"dateTime":"29-11-2024 14:30:00"}
TZ	符合常見時區縮寫 (PST、PDT、MST、MDT、CST CDT、EST、EDT、UTC)。	20	輸入： PDT 模式：%{TZ:tz} 輸出： {"tz":"PDT"}

Grok 模式	Description	模式上限	範例
DATESTAMP_RFC822	符合格式的日期和時間：Day MonthName (D)D (YY)YY (H)H：mm：(s)s 時區	20	輸入：`Monday Jan 5 23 1:30:00 CDT` 模式：`%{DATESTAMP_RFC822:dateTime}` 輸出：`{"dateTime":"Monday Jan 5 23 1:30:00 CDT"}` 輸入：`Mon January 15 2023 14:30:00 PST` 模式：`%{DATESTAMP_RFC822:dateTime}` 輸出：`{"dateTime":"Mon January 15 2023 14:30:00 PST"}`

Grok 模式	Description	模式上限	範例
DATESTAMP_RFC2822	符合 RFC2822 日期時間格式：Day，(d)d MonthName (yy)yy (H)H：mm：(s)s Z|【+-】(H)H：mm	20	輸入：`Mon, 15 May 2023 14:30:00 +0530` 模式：`%{DATESTAMP_RFC2822:dateTime}` 輸出：`{"dateTime":"Mon, 15 May 2023 14:30:00 +0530"}` 輸入：`Monday, 15 Jan 23 14:30:00 Z` 模式：`%{DATESTAMP_RFC2822:dateTime}` 輸出：`{"dateTime":"Monday, 15 Jan 23 14:30:00 Z"}`
DATESTAMP_OTHER	符合格式的日期和時間：Day MonthName (d)d (H)H：mm：(s)s Timezone (yy)yy	20	輸入：`Mon May 15 14:30:00 PST 2023` 模式：`%{DATESTAMP_OTHER:dateTime}` 輸出：`{"dateTime":"Mon May 15 14:30:00 PST 2023"}`

Grok 模式	Description	模式上限	範例
DATESTAMP_EVENTLOG	符合不含分隔符號的精簡日期時間格式：(yy)yyMM(d)d(H)Hmm(s)s	20	輸入： 20230515143000 模式：%{DATESTAMP_EVENTLOG:dateTime} 輸出： {"dateTime":"20230515143000"}

日誌 grok 模式

Grok 模式	Description	模式上限	範例
LOGLEVEL	符合不同大寫和縮寫中的標準日誌層級，包括下列項目：Alert/ALERT 、Trace/TRACE 、Debug/DEBUG 、Notice/NOTICE Info/INFO 、Warn/Warning/WARN/WARNING 、Err/Error/ERR/ERROR 、Crit/Critical/CRIT/CRITICAL 、Fatal/FATAL 、、Severe/SEVERE Emerg/Emergency/EMERG/EMERGENCY	20	輸入： INFO 模式：%{LOGLEVEL:logLevel} 輸出： {"logLevel":"INFO"}
HTTPDATE	符合日誌檔案中常用的日期和時間格式。格式：(d)d/MonthName/(yy)yy：	20	輸入： 23/Nov/2024:14:30:00 +0640

Grok 模式	Description	模式上限	範例
	(H)H：mm：(s)s 時區 MonthName： 符合完整或縮寫的英文月份名稱（範例：「Jan」或「January」) 時區： 符合 %{INT} grok 模式		模式：%{HTTPDATE:date} 輸出： {"date":"23/Nov/2024:14:30:00 +0640"}
SYSLOGTIMESTAMP	與 MonthName (d)d (H)H：mm：(s)s MonthName 相符的日期格式：符合完整或縮寫的英文月份名稱（範例：「1 月」或「1 月」)	20	輸入： Nov 29 14:30:00 模式：%{SYSLOGTIMESTAMP:dateTime} 輸出： {"dateTime":"Nov 29 14:30:00"}
PROG	符合由字母、數字、點、底線、正斜線、百分比符號和連字號字元組成的程式名稱。	20	輸入： user.profile/settings-page 模式：%{PROG:program} 輸出： {"program":"user.profile/settings-page"}

Grok 模式	Description	模式上限	範例
SYSLOGPRO G	選擇性地比對 PROG grok 模式，後面接著方括號中的程序 ID。	20	輸入： `user.profile/settings-page[1234]` 模式：`%{SYSLOGPROG:programWithId}` 輸出： `{"programWithId":"user.profile/settings-page[1234]","program":"user.profile/settings-page","pid":"1234"}`
SYSLOGHOST	符合 %{HOST} 或 %{IP} 模式	5	輸入： `2001:db8:3333:4444:5555:6666:7777:8888` 模式：`%{SYSLOGHOST:ip}` 輸出： `{"ip":"2001:db8:3333:4444:5555:6666:7777:8888"}`

Grok 模式	Description	模式上限	範例
SYSLOGFAC ILITY	符合十進位格式的 syslog 優先順序。該值應該用角括號括住 (<>)。	20	輸入： <13.6> 模式：%{SYSLOGF ACILITY:syslog} 輸出： {"syslog" :"<13.6>" ,"facilit y":"13"," priority":"6"}

常見日誌 grok 模式

您可以使用預先定義的自訂 grok 模式來比對 Apache、NGINX 和 Syslog Protocol (RFC 5424) 日誌格式。當您使用這些特定模式時，它們必須是相符組態中的第一個模式，而且前面沒有其他模式。此外，您只能使用一個 DATA 來追蹤它們。GREEDYDATA 或 GREEDYDATA_MULTILINE 模式。

Grok 模式	Description	模式上限
APACHE_ACCESS_LOG	符合 Apache 存取日誌	1
NGINX_ACCESS_LOG	符合 NGINX 存取日誌	1
SYSLOG5424	符合 Syslog 通訊協定 (RFC 5424) 日誌	1

以下顯示使用這些常見日誌格式模式的有效和無效範例。

```
"%{NGINX_ACCESS_LOG} %{DATA}" // Valid
"%{SYSLOG5424}%{DATA:logMsg}" // Valid
"%{APACHE_ACCESS_LOG} %{GREEDYDATA:logMsg}" // Valid
"%{APACHE_ACCESS_LOG} %{SYSLOG5424}" // Invalid (multiple common log patterns used)
```

```
"%{NGINX_ACCESS_LOG} %{NUMBER:num}" // Invalid (Only GREEDYDATA and DATA patterns are
 supported with common log patterns)
"%{GREEDYDATA:logMsg} %{SYSLOG5424}" // Invalid (GREEDYDATA and DATA patterns are
 supported only after common log patterns)
```

常見日誌格式範例

Apache 日誌範例

範例日誌：

```
127.0.0.1 - - [03/Aug/2023:12:34:56 +0000] "GET /page.html HTTP/1.1" 200 1234
```

轉換器：

```
[
     {
        "grok": {
            "match": "%{APACHE_ACCESS_LOG}"
        }
    }
]
```

輸出：

```
{
    "request": "/page.html",
    "http_method": "GET",
    "status_code": 200,
    "http_version": "1.1",
    "response_size": 1234,
    "remote_host": "127.0.0.1",
    "timestamp": "2023-08-03T12:34:56Z"
}
```

NGINX 日誌範例

範例日誌：

```
192.168.1.100 - Foo [03/Aug/2023:12:34:56 +0000] "GET /account/login.html HTTP/1.1"
 200 42 "https://www.amazon.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36"
```

轉換器：

```
[
     {
        "grok": {
            "match": "%{NGINX_ACCESS_LOG}"
        }
    }
]
```

輸出：

```
{
    "request": "/account/login.html",
    "referrer": "https://www.amazon.com/",
    "agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/92.0.4515.131 Safari/537.36",
    "http_method": "GET",
    "status_code": 200,
    "auth_user": "Foo",
    "http_version": "1.1",
    "response_size": 42,
    "remote_host": "192.168.1.100",
    "timestamp": "2023-08-03T12:34:56Z"
}
```

Syslog 通訊協定 (RFC 5424) 日誌範例

範例日誌：

```
<165>1 2003-10-11T22:14:15.003Z mymachine.example.com evntslog - ID47
 [exampleSDID@32473 iut="3" eventSource= "Application" eventID="1011"]
[examplePriority@32473 class="high"]
```

轉換器：

```
[
     {
        "grok": {
            "match": "%{SYSLOG5424}"
        }
```

```
    }
]
```

輸出：

```
{
  "pri": 165,
  "version": 1,
  "timestamp": "2003-10-11T22:14:15.003Z",
  "hostname": "mymachine.example.com",
  "app": "evntslog",
  "msg_id": "ID47",
  "structured_data": "exampleSDID@32473 iut=\"3\" eventSource= \"Application\" eventID=
\"1011\"",
  "message": "[examplePriority@32473 class=\"high\"]"
}
```

CSV

csv 處理器會將逗號分隔值 (CSV) 從日誌事件剖析為資料欄。

欄位	Description	是否為必要？	預設	限制
source	要剖析之日誌事件中 欄位的路徑	否	`@message`	長度上限：128 巢狀金鑰深度上限：3
分隔符號	用來分隔原始逗號分隔值日誌事件中每個資料欄的字元	否	`,`	長度上限：1，除非值為 `\t`或 `\s`
quoteCharacter	用作單一資料欄文字限定詞的字元	否	`"`	長度上限：1
欄	要用於轉換日誌事件中資料欄的名稱清單。	否	`[column_1, column_2`	CSV 資料欄上限：100 長度上限：128 巢狀金鑰深度上限：3

欄位	Description	是否為必要？	預設	限制
目的地	要在 下放置轉換金鑰值對的父欄位	否	Root node	長度上限：128 巢狀金鑰深度上限：3

delimiter 將 設定為 \t會將標籤字元上的每個資料欄分開，並將單一空格字元上的每個資料欄\t分開。

範例

假設部分擷取的日誌事件如下所示：

```
'Akua Mansa':28:'New York: USA'
```

假設我們僅使用 csv 處理器：

```
[
     {
        "csv": {
            "delimiter": ":",
            "quoteCharacter": "'"
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "column_1": "Akua Mansa",
  "column_2": "28",
  "column_3": "New York: USA"
}
```

範例 2

假設擷取的日誌事件如下所示：

```
{
```

```
    "timestamp": "2024-11-23T16:03:12Z",
    "type": "user_data",
    "logMsg": "'Akua Mansa':28:'New York: USA'"
}
```

假設我們將事件剖析為 JSON，它們會使用 csv 處理器剖析 JSON 欄位，並指定資料欄名稱和目的地：

```
[
    {
        "parseJSON": {}
    },
    {
        "csv": {
            "source": "logMsg",
            "delimiter": ":",
            "quoteCharacter": "'",
            "columns":["name","age","location"],
            "destination": "msg"
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "timestamp": "2024-11-23T16:03:12Z",
    "logMsg": "'Akua Mansa':28:'New York: USA'",
    "type": "user_data",
    "msg": {
        "name": "Akua Mansa",
        "age": "28",
        "location": "New York: USA"
    }
}
```

parseKeyValue

使用 parseKeyValue 處理器將指定的欄位剖析為鍵/值對。您可以使用下列選項自訂處理器來剖析欄位資訊。

欄位	Description	是否為必要？	預設	限制
source	要剖析之日誌事件中 欄位的路徑	否	`@message`	長度上限：128 巢狀金鑰深度上限：3
目的地	要將擷取的鍵/值對放入其中的目的地欄位	否		長度上限：128
fieldDelimiter	在原始日誌事件的鍵值對之間使用的欄位分隔符號字串	否	&	長度上限：128
keyValueDelimiter	在轉換日誌事件中每對索引鍵和值之間使用的分隔符號字串	否	=	長度上限：128
nonMatchValue	當鍵/值對未成功分割時，要插入結果中值欄位的值。	否		長度上限：128
keyPrefix	如果您想要新增字首所有轉換的金鑰，請在此處指定它。	否		長度上限：128
overwriteIfExists	如果目的地金鑰已存在，是否覆寫該值	否	`false`	

範例

採用下列範例日誌事件：

```
key1:value1!key2:value2!key3:value3!key4
```

假設我們使用下列處理器組態：

```
[
    {
        "parseKeyValue": {
            "destination": "new_key",
            "fieldDelimiter": "!",
            "keyValueDelimiter": ":",
```

```
            "nonMatchValue": "defaultValue",
            "keyPrefix": "parsed_"
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "new_key": {
    "parsed_key1": "value1",
    "parsed_key2": "value2",
    "parsed_key3": "value3",
    "parsed_key4": "defaultValue"
  }
}
```

適用於 AWS 付費日誌的內建處理器

本節包含有關內建處理器的資訊，您可以將這些處理器與提供日誌 AWS 的服務搭配使用。

內容

parseWAF

使用此處理器來剖析已 AWS WAF 結束的日誌，它會採用 的內容，`httpRequest.headers`並從每個標頭名稱建立具有對應值的 JSON 金鑰。它也會對 執行相同的操作`labels`。這些轉換可讓您更輕鬆地查詢 AWS WAF 日誌。如需 AWS WAF 日誌格式的詳細資訊，請參閱 Web ACL 流量的日誌範例。

此處理器僅接受 `@message`做為輸入。

> **Important**
>
> 如果您使用此處理器，它必須是轉換器中的第一個處理器。

範例

採用下列範例日誌事件：

```
{
  "timestamp": 1576280412771,
  "formatVersion": 1,
  "webaclId": "arn:aws:wafv2:ap-southeast-2:111122223333:regional/webacl/
STMTest/1EXAMPLE-2ARN-3ARN-4ARN-123456EXAMPLE",
  "terminatingRuleId": "STMTest_SQLi_XSS",
  "terminatingRuleType": "REGULAR",
  "action": "BLOCK",
  "terminatingRuleMatchDetails": [
    {
      "conditionType": "SQL_INJECTION",
      "sensitivityLevel": "HIGH",
      "location": "HEADER",
      "matchedData": ["10", "AND", "1"]
    }
  ],
  "httpSourceName": "-",
  "httpSourceId": "-",
  "ruleGroupList": [],
  "rateBasedRuleList": [],
  "nonTerminatingMatchingRules": [],
  "httpRequest": {
    "clientIp": "1.1.1.1",
    "country": "AU",
    "headers": [
      { "name": "Host", "value": "localhost:1989" },
      { "name": "User-Agent", "value": "curl/7.61.1" },
      { "name": "Accept", "value": "*/*" },
      { "name": "x-stm-test", "value": "10 AND 1=1" }
    ],
    "uri": "/myUri",
    "args": "",
    "httpVersion": "HTTP/1.1",
    "httpMethod": "GET",
```

```
    "requestId": "rid"
  },
  "labels": [{ "name": "value" }]
}
```

處理器組態如下：

```
[
    {
        "parseWAF": {}
    }
]
```

轉換的日誌事件如下。

```
{
  "httpRequest": {
    "headers": {
      "Host": "localhost:1989",
      "User-Agent": "curl/7.61.1",
      "Accept": "*/*",
      "x-stm-test": "10 AND 1=1"
    },
    "clientIp": "1.1.1.1",
    "country": "AU",
    "uri": "/myUri",
    "args": "",
    "httpVersion": "HTTP/1.1",
    "httpMethod": "GET",
    "requestId": "rid"
  },
  "labels": { "name": "value" },
  "timestamp": 1576280412771,
  "formatVersion": 1,
  "webaclId": "arn:aws:wafv2:ap-southeast-2:111122223333:regional/webacl/
STMTest/1EXAMPLE-2ARN-3ARN-4ARN-123456EXAMPLE",
  "terminatingRuleId": "STMTest_SQLi_XSS",
  "terminatingRuleType": "REGULAR",
  "action": "BLOCK",
  "terminatingRuleMatchDetails": [
    {
      "conditionType": "SQL_INJECTION",
      "sensitivityLevel": "HIGH",
```

```
      "location": "HEADER",
      "matchedData": ["10", "AND", "1"]
    }
  ],
  "httpSourceName": "-",
  "httpSourceId": "-",
  "ruleGroupList": [],
  "rateBasedRuleList": [],
  "nonTerminatingMatchingRules": []
}
```

parsePostgres

使用此處理器來剖析 Amazon RDS for PostgreSQL 付費日誌、擷取欄位，並將它們轉換為 JSON 格式。如需 RDS for PostgreSQL 日誌格式的詳細資訊，請參閱 RDS for PostgreSQL 資料庫日誌檔案。

此處理器僅接受 `@message`做為輸入。

> **Important**
>
> 如果您使用此處理器，它必須是轉換器中的第一個處理器。

範例

採用下列範例日誌事件：

```
2019-03-10 03:54:59 UTC:10.0.0.123(52834):postgres@logtestdb:[20175]:ERROR: column
 "wrong_column_name" does not exist at character 8
```

處理器組態如下：

```
[
    {
        "parsePostgres": {}
    }
]
```

轉換的日誌事件如下。

```
{
```

```
  "logTime": "2019-03-10 03:54:59 UTC",
  "srcIp": "10.0.0.123(52834)",
  "userName": "postgres",
  "dbName": "logtestdb",
  "processId": "20175",
  "logLevel": "ERROR"
}
```

parseCloudfront

使用此處理器來剖析 Amazon CloudFront 付費日誌、擷取欄位，並將它們轉換為 JSON 格式。編碼的欄位值會解碼。整數和雙數的值會被視為這樣。如需 Amazon CloudFront 日誌格式的詳細資訊，請參閱設定和使用標準日誌（存取日誌）。

此處理器僅接受 @message做為輸入。

> **Important**
>
> 如果您使用此處理器，它必須是轉換器中的第一個處理器。

範例

採用下列範例日誌事件：

```
2019-12-04  21:02:31   LAX1   392    192.0.2.24    GET
 d111111abcdef8.cloudfront.net  /index.html    200    -  Mozilla/5.0%20(Windows
%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36  -  -  Hit
    SOX4xwn4XV6Q4rgb7XiVGOHms_BGlTAC4KyHmureZmBNrjGdRLiNIQ==
 d111111abcdef8.cloudfront.net  https  23 0.001  -  TLSv1.2    ECDHE-RSA-AES128-GCM-
SHA256    Hit    HTTP/2.0   -  -  11040  0.001  Hit    text/html  78 -  -
```

處理器組態如下：

```
[
    {
        "parseCloudfront": {}
    }
]
```

轉換的日誌事件如下。

```
{
  "date": "2019-12-04",
  "time": "21:02:31",
  "x-edge-location": "LAX1",
  "sc-bytes": 392,
  "c-ip": "192.0.2.24",
  "cs-method": "GET",
  "cs(Host)": "d111111abcdef8.cloudfront.net",
  "cs-uri-stem": "/index.html",
  "sc-status": 200,
  "cs(Referer)": "-",
  "cs(User-Agent)": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36",
  "cs-uri-query": "-",
  "cs(Cookie)": "-",
  "x-edge-result-type": "Hit",
  "x-edge-request-id": "SOX4xwn4XV6Q4rgb7XiVGOHms_BGlTAC4KyHmureZmBNrjGdRLiNIQ==",
  "x-host-header": "d111111abcdef8.cloudfront.net",
  "cs-protocol": "https",
  "cs-bytes": 23,
  "time-taken": 0.001,
  "x-forwarded-for": "-",
  "ssl-protocol": "TLSv1.2",
  "ssl-cipher": "ECDHE-RSA-AES128-GCM-SHA256",
  "x-edge-response-result-type": "Hit",
  "cs-protocol-version": "HTTP/2.0",
  "fle-status": "-",
  "fle-encrypted-fields": "-",
  "c-port": 11040,
  "time-to-first-byte": 0.001,
  "x-edge-detailed-result-type": "Hit",
  "sc-content-type": "text/html",
  "sc-content-len": 78,
  "sc-range-start": "-",
  "sc-range-end": "-"
}
```

parseRoute53

使用此處理器來剖析 Amazon Route 53 Public Data Plane 付費日誌、擷取欄位，並將它們轉換為 JSON 格式。編碼的欄位值會解碼。此處理器不支援 Amazon Route 53 Resolver 日誌。

此處理器僅接受 @message做為輸入。

> **Important**
>
> 如果您使用此處理器，它必須是轉換器中的第一個處理器。

範例

採用下列範例日誌事件：

```
1.0 2017-12-13T08:15:50.235Z Z123412341234 example.com AAAA NOERROR TCP IAD12 192.0.2.0
 198.51.100.0/24
```

處理器組態如下：

```
[
    {
        "parseRoute53": {}
    }
]
```

轉換的日誌事件如下。

```
{
  "version": 1.0,
  "queryTimestamp": "2017-12-13T08:15:50.235Z",
  "hostZoneId": "Z123412341234",
  "queryName": "example.com",
  "queryType": "AAAA",
  "responseCode": "NOERROR",
  "protocol": "TCP",
  "edgeLocation": "IAD12",
  "resolverIp": "192.0.2.0",
  "ednsClientSubnet": "198.51.100.0/24"
}
```

parseVPC

使用此處理器剖析 Amazon VPC 提供的日誌、擷取欄位，並將它們轉換為 JSON 格式。編碼的欄位值會解碼。

此處理器僅接受 @message做為輸入。

> **Important**
>
> 如果您使用此處理器，它必須是轉換器中的第一個處理器。

範例

採用下列範例日誌事件：

```
2 123456789010 eni-abc123de 192.0.2.0 192.0.2.24 20641 22 6 20 4249 1418530010
 1418530070 ACCEPT OK
```

處理器組態如下：

```
[
    {
        "parseVPC": {}
    }
]
```

轉換的日誌事件如下。

```
{
  "version": 2,
  "accountId": "123456789010",
  "interfaceId": "eni-abc123de",
  "srcAddr": "192.0.2.0",
  "dstAddr": "192.0.2.24",
  "srcPort": 20641,
  "dstPort": 22,
  "protocol": 6,
  "packets": 20,
  "bytes": 4249,
  "start": 1418530010,
  "end": 1418530070,
  "action": "ACCEPT",
  "logStatus": "OK"
}
```

parseToOCSF

parseToOCSF 處理器會將日誌轉換為開放網路安全結構描述架構 (OCSF) 事件。OCSF 是一項開放標準，提供常見的安全資料結構描述，可在不同的安全工具和平台上提供更好的互通性和分析。

此處理器特別適用於您需要將各種 AWS 服務的日誌格式標準化為一致結構描述以進行下游分析的安全分析工作流程。

參數

eventSource (必要)

指定產生要轉換之日誌事件 AWS 的服務或程序。有效的值如下：

- CloudTrail - CloudTrail 日誌
- Route53Resolver - Route 53 Resolver 日誌
- VPCFlow - Amazon VPC 流程日誌
- EKSAudit - Amazon EKS 稽核日誌
- AWSWAF - AWS WAF logs

ocsfVersion (必要)

指定要用於轉換日誌事件的 OCSF 結構描述版本。目前支援的版本： V1.1, V1.5

mappingVersion (選用)

指定 OCSF 轉換映射版本。控制將日誌轉換為 OCSF 格式時套用哪些轉換邏輯。如果未指定， 會在政策建立時使用最新的可用版本。發佈新的映射版本時，現有的政策不會自動升級。目前最新版本：v1.5.0。

注意：當 ocsfVersion為 時，不支援 V1.1。

source (選用)

您要剖析之日誌事件中 欄位的路徑。如果省略，則會剖析整個日誌訊息。

範例

下列範例示範如何使用 parseToOCSF將 VPC 流程日誌轉換為 OCSF 格式：

```
{
  "parseToOCSF": {
    "eventSource": "VPCFlow",
```

```
    "ocsfVersion": "V1.1"
  }
}
```

下列範例示範如何指定特定映射版本，以實現一致的轉換行為：

```
{
  "parseToOCSF": {
    "eventSource": "CloudTrail",
    "ocsfVersion": "V1.5",
    "mappingVersion": "v1.5.0"
  }
}
```

字串變動處理器

本節包含您可以與日誌事件轉換器搭配使用的字串變動處理器相關資訊。

內容

lowerCaseString

`lowerCaseString` 處理器會將字串轉換為小寫版本。

欄位	Description	是否為必要？	預設	限制
withKeys	要轉換為小寫的索引鍵清單	是		項目上限：10

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "INNER_VALUE"
    }
}
```

轉換器組態是這樣，使用 lowerCaseString搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "lowerCaseString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

upperCaseString

upperCaseString 處理器會將字串轉換為其大寫版本。

欄位	Description	是否為必要？	預設	限制
withKeys	要轉換為大寫的金鑰清單	是		項目上限：10

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是這樣，使用 upperCaseString搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "upperCaseString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": "INNER_VALUE"
  }
}
```

splitString

splitString 處理器是一種字串變動處理器，使用分隔字元將欄位分割為陣列。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都必須包含 source和 delimiter 欄位。	是		項目上限：10
source	要分割之欄位值的索引鍵	是		長度上限：128

欄位	Description	是否為必要？	預設	限制
分隔符號	要在 上分割欄位值的分隔符號字串	是		長度上限：128

範例 1

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是如此，使用 splitString搭配 parseJSON：

```
[
     {
        "parseJSON": {}
    },
    {
         "splitString": {
            "entries": [
                {
                    "source": "outer_key.inner_key",
                    "delimiter": "_"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": [
      "inner",
      "value"
    ]
```

```
  }
}
```

範例 2

要在 上分割字串的分隔符號長度可以是多個字元。

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "item1, item2, item3"
    }
}
```

轉換器組態如下所示：

```
[
     {
        "parseJSON": {}
    },
    {
         "splitString": {
            "entries": [
                {
                    "source": "outer_key.inner_key",
                    "delimiter": ", "
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": [
      "item1",
      "item2",
      "item3"
    ]
```

```
  }
}
```

substituteString

substituteString 處理器是一種字串變動處理器，會將索引鍵的值與規則表達式比對，並以取代字串取代所有比對。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目必須包含 source、 from和 to 欄位。	是		項目上限：10
source	要修改的欄位索引鍵	是		長度上限：128 巢狀金鑰深度上限：3
from	要取代的規則表達式字串。使用雙引號時必須使用 \\ 逸出 【 和 】 等特殊規則運算式字元，使用單引號時或從 設定時必須使用 \ 逸出 AWS 管理主控台。如需詳細資訊，請參閱 Oracle 網站上的類別模式。 您可以在 中包裝模式(...)，以建立編號的擷取群組，並建立可在 to 欄位中參考的(?P<group_name>...) 具名擷取群組。	是		長度上限：128
至	對於擷取群組的每個 from Backreference 配對，可以使用要取代的字串。將 $n 格式用於編號群組，例如 $1，並將 ${group_name} 用於具名群組，例如 ${my_group} 。>	是		長度上限：128 最大反引數數：10 重複的反向參考數目上限：2

範例 1

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key1": "[]",
        "inner_key2": "123-345-567",
        "inner_key3": "A cat takes a catnap."
    }
}
```

轉換器組態是如此，使用 substituteString搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "substituteString": {
            "entries": [
                {
                    "source": "outer_key.inner_key1",
                    "from": "\\[\\]",
                    "to": "value1"
                },
                {
                    "source": "outer_key.inner_key2",
                    "from": "[0-9]{3}-[0-9]{3}-[0-9]{3}",
                    "to": "xxx-xxx-xxx"
                },
                {
                    "source": "outer_key.inner_key3",
                    "from": "cat",
                    "to": "dog"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
```

```
    "inner_key1": "value1",
    "inner_key2": "xxx-xxx-xxx",
    "inner_key3": "A dog takes a dognap."
  }
}
```

範例 2

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key1": "Tom, Dick, and Harry",
        "inner_key2": "arn:aws:sts::123456789012:assumed-role/MyImportantRole/
MySession"
    }
}
```

轉換器組態是如此，使用 substituteString搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "substituteString": {
            "entries": [
                {
                    "source": "outer_key.inner_key1",
                    "from": "(\w+), (\w+), and (\w+)",
                    "to": "$1 and $3"
                },
                {
                    "source": "outer_key.inner_key2",
                    "from": "^arn:aws:sts::(?P<account_id>\\d{12}):assumed-role/(?
P<role_name>[\\w+=,.@-]+)/(?P<role_session_name>[\\w+=,.@-]+)$",
                    "to": "${account_id}:${role_name}:${role_session_name}"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key1": "Tom and Harry",
    "inner_key2": "123456789012:MyImportantRole:MySession"
  }
}
```

trimString

`trimString` 處理器會從金鑰的開頭和結尾移除空格。

欄位	Description	是否為必要？	預設	限制
withKeys	要修剪的金鑰清單	是		項目上限：10

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "   inner_value  "
    }
}
```

轉換器組態是如此，使用 `trimString`搭配 `parseJSON`：

```
[
    {
        "parseJSON": {}
    },
    {
        "trimString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

JSON 變動處理器

本節包含您可以與日誌事件轉換器搭配使用的 JSON 變動處理器相關資訊。

內容

addKeys

使用addKeys處理器將新的鍵/值對新增至日誌事件。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都可以包含 key、 value和 overwriteIfExists 欄位。	是		項目上限：5
金鑰	要新增之新項目的索引鍵	是		長度上限：128 巢狀金鑰深度上限：3
value	要新增的新項目值	是		長度上限：256

欄位	Description	是否為必要？	預設	限制
overwriteIfExists	如果您將此設定為 `true`，則如果事件中key已存在，則會覆寫現有的值。預設值為 `false`。	否	false	沒有限制

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是這樣，使用 addKeys搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "addKeys": {
            "entries": [
                {
                    "key": "outer_key.new_key",
                    "value": "new_value"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": "inner_value",
```

```
    "new_key": "new_value"
  }
}
```

deleteKeys

使用deleteKeys處理器從日誌事件中刪除欄位。這些欄位可以包含鍵/值對。

欄位	Description	是否為必要？	預設	限制
withKeys	要刪除的金鑰清單。	是	沒有限制	項目上限：5

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是這樣，使用 deleteKeys搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "deleteKeys": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {}
```

```
}
```

moveKeys

使用moveKeys處理器將金鑰從一個欄位移至另一個欄位。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都可以包含 source、 target和 overwriteIfExists 欄位。	是		項目上限：5
source	要移動的金鑰	是		長度上限：128 巢狀金鑰深度上限：3
目標	要移至的金鑰	是		長度上限：128 巢狀金鑰深度上限：3
overwriteIfExists	如果您將此設定為 true，則如果事件中key已存在，則會覆寫現有的值。預設值為 false。	否	false	沒有限制

範例

採用下列範例日誌事件：

```
{
    "outer_key1": {
        "inner_key1": "inner_value1"
    },
    "outer_key2": {
        "inner_key2": "inner_value2"
    }
}
```

轉換器組態是這樣，使用 moveKeys搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "moveKeys": {
            "entries": [
                {
                    "source": "outer_key1.inner_key1",
                    "target": "outer_key2"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key1": {},
  "outer_key2": {
    "inner_key2": "inner_value2",
    "inner_key1": "inner_value1"
  }
}
```

renameKeys

使用`renameKeys`處理器重新命名日誌事件中的金鑰。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都可以包含 `key`、 `target`和 `overwriteIfExists` 欄位。	是	沒有限制	項目上限：5
金鑰	要重新命名的金鑰	是	沒有限制	長度上限：128
目標	新的金鑰名稱	是	沒有限制	長度上限：128

欄位	Description	是否為必要？	預設	限制
				巢狀金鑰深度上限：3
overwriteIfExists	如果您將此設定為 `true`，則如果事件中key已存在，則會覆寫現有的值。預設值為 `false`。	否	false	沒有限制

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是這樣，使用 `renameKeys`搭配 `parseJSON`：

```
[
    {
        "parseJSON": {}
    },
    {
        "renameKeys": {
            "entries": [
                {
                    "key": "outer_key",
                    "target": "new_key"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "new_key": {
```

```
    "inner_key": "inner_value"
  }
}
```

copyValue

使用copyValue處理器在日誌事件中複製值。您也可以使用此處理器，
透過將下列中繼資料金鑰的值複製到日誌事件，將中繼資料新增至日誌事
件：@logGroupName、@logGroupStream、@accountId、@regionName。這會在下列範例中說明。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都可以包含 source、 target和 overwriteIfExists 欄位。	是		項目上限：5
source	要複製的金鑰	是		長度上限：128 巢狀金鑰深度上限：3
目標	將值複製到 的金鑰	是	沒有限制	長度上限：128 巢狀金鑰深度上限：3
overwriteIfExists	如果您將此設定為 true，則如果事件中key已存在，則會覆寫現有的值。預設值為 false。	否	false	沒有限制

範例

採用下列範例日誌事件：

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

轉換器組態是這樣，使用 copyValue搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "copyValue": {
            "entries": [
                {
                    "key": "outer_key.new_key",
                    "target": "new_key"
                },
                {
                    "source": "@logGroupName",
                    "target": "log_group_name"
                },
                {
                    "source": "@logGroupStream",
                    "target": "log_group_stream"
                },
                {
                    "source": "@accountId",
                    "target": "account_id"
                },
                {
                    "source": "@regionName",
                    "target": "region_name"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
  "outer_key": {
    "inner_key": "inner_value"
  },
  "new_key": "inner_value",
  "log_group_name": "myLogGroupName",
  "log_group_stream": "myLogStreamName",
```

```
  "account_id": "012345678912",
  "region_name": "us-east-1"
}
```

listToMap

listToMap 處理器會取得包含金鑰欄位的物件清單，並將其轉換為目標金鑰的映射。

欄位	Description	是否為必要？	預設	限制
source	ProcessingEvent 中的金鑰，其中包含將轉換為映射的物件清單	是		長度上限：128 巢狀金鑰深度上限：3
金鑰	要擷取為所產生映射中索引鍵的欄位索引鍵	是		長度上限：128
valueKey	如果指定此值，您在此參數中指定的值將從source物件中擷取，並放入產生的映射值中。否則，來源清單中的原始物件會放入產生的映射值中。	否		長度上限：128
目標	將存放所產生映射之欄位的索引鍵	否	根節點	長度上限：128 巢狀金鑰深度上限：3
flatten	布林值，指出清單是否將扁平化為單一項目，或產生的映射中的值是否將是清單。 根據預設，相符索引鍵的值會以陣列表示。flatten 設定為 true 以根據 的值將陣列轉換為單一值flattenedElement 。	否	false	
flattened Element	如果您將 flatten設定為 true，請使用 flattenedElement 指定要保留的元素 lastfirst或 。	flatten 當 設定為		值只能為 first或 last

欄位	Description	是否為必要？	預設	限制
		時為必要 `true`		

範例

採用下列範例日誌事件：

```
{
    "outer_key": [
        {
            "inner_key": "a",
            "inner_value": "val-a"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b1"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b2"
        },
        {
            "inner_key": "c",
            "inner_value": "val-c"
        }
    ]
}
```

使用案例 1 的轉換器： `flatten`是 `false`

```
[
    {
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
```

```
            "valueKey": "inner_value",
            "flatten": false
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "outer_key": [
        {
            "inner_key": "a",
            "inner_value": "val-a"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b1"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b2"
        },
        {
            "inner_key": "c",
            "inner_value": "val-c"
        }
    ],
    "a": [
        "val-a"
    ],
    "b": [
        "val-b1",
        "val-b2"
    ],
    "c": [
        "val-c"
    ]
}
```

使用案例 2 的轉換器： `flatten` 是 `true`， `flattenedElement`是 `first`

```
[
    {
```

```
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
            "valueKey": "inner_value",
            "flatten": true,
            "flattenedElement": "first"
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "outer_key": [
        {
            "inner_key": "a",
            "inner_value": "val-a"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b1"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b2"
        },
        {
            "inner_key": "c",
            "inner_value": "val-c"
        }
    ],
    "a": "val-a",
    "b": "val-b1",
    "c": "val-c"
}
```

使用案例 3 的轉換器： `flatten` 是 `true`， `flattenedElement`是 `last`

```
[
    {
```

```
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
            "valueKey": "inner_value",
            "flatten": true,
            "flattenedElement": "last"
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "outer_key": [
        {
            "inner_key": "a",
            "inner_value": "val-a"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b1"
        },
        {
            "inner_key": "b",
            "inner_value": "val-b2"
        },
        {
            "inner_key": "c",
            "inner_value": "val-c"
        }
    ],
    "a": "val-a",
    "b": "val-b2",
    "c": "val-c"
}
```

資料類型轉換器處理器

本節包含您可以與日誌事件轉換器搭配使用之資料類型轉換器處理器的相關資訊。

內容

typeConverter

使用typeConverter處理器將與指定索引鍵相關聯的值類型轉換為指定的類型。它是一種轉換處理器，可變更指定欄位的類型。值可以轉換為下列其中一種資料類型：integer、 doublestring和boolean。

欄位	Description	是否為必要？	預設	限制
項目	項目陣列。陣列中的每個項目都必須包含 key和 type 欄位。	是		項目上限：10
金鑰	具有要轉換為不同類型之值的金鑰	是		長度上限：128 巢狀金鑰深度上限：3
type	要轉換的類型。有效值為 integer、 doublestring和boolean。	是		

範例

採用下列範例日誌事件：

```
{
    "name": "value",
    "status": "200"
}
```

轉換器組態是這樣，使用 typeConverter搭配 parseJSON：

```
[
    {
```

```
        "parseJSON": {}
    },
    {
        "typeConverter": {
            "entries": [
                {
                    "key": "status",
                    "type": "integer"
                }
            ]
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "name": "value",
    "status": 200
}
```

datetimeConverter

使用`datetimeConverter`處理器將日期時間字串轉換為您指定的格式。

欄位	Description	是否為必要？	預設	限制
source	要套用日期轉換的金鑰。	是		項目上限：10
matchPatterns	要比對 `source` 欄位的模式清單	是		項目上限：5
目標	存放結果的 JSON 欄位。	是		長度上限：128 巢狀金鑰深度上限：3
targetFormat	用於目標欄位中轉換資料的日期時間格式。	否	`yyyy-MM-dd'T'HH:mm`	長度上限：64

欄位	Description	是否為必要？	預設	限制
			:ss.SSS'Z	
sourceTimezone	來源欄位的時區。 如需可能值的清單，請參閱 Java 支援的區域 ID 和位移。	否	UTC	長度下限：1
targetTimezone	目標欄位的時區。 如需可能值的清單，請參閱 Java 支援的區域 ID 和位移。	否	UTC	長度下限：1
locale	來源欄位的地區設定。 如需可能值的清單，請參閱 Java 中的 Locale getAvailableLocales() 方法與範例。	是		長度下限：1

範例

採用下列範例日誌事件：

```
{"german_datetime": "Samstag 05. Dezember 1998 11:00:00"}
```

轉換器組態是這樣，使用 dateTimeConverter搭配 parseJSON：

```
[
    {
        "parseJSON": {}
    },
    {
        "dateTimeConverter": {
            "source": "german_datetime",
            "target": "target_1",
            "locale": "de",
            "matchPatterns": ["EEEE dd. MMMM yyyy HH:mm:ss"],
            "sourceTimezone": "Europe/Berlin",
```

```
            "targetTimezone": "America/New_York",
            "targetFormat": "yyyy-MM-dd'T'HH:mm:ss z"
        }
    }
]
```

轉換的日誌事件如下。

```
{
    "german_datetime": "Samstag 05. Dezember 1998 11:00:00",
    "target_1": "1998-12-05T17:00:00 MEZ"
}
```

轉換指標和錯誤

CloudWatch Logs 會將轉換指標發佈至 CloudWatch。這些指標包括 `TransformedLogEvents`、`TransformedBytes`和 `TransformationErrors`。如需詳細資訊，請參閱日誌轉換器指標和維度。

每當 CloudWatch Logs 嘗試轉換日誌事件而失敗時，就會將`@transformationError`系統欄位新增至該日誌事件。當您執行 CloudWatch Logs Insights 查詢時，您會在所有發生轉換失敗的日誌事件中看到此欄位。您可以使用 等查詢來查詢此欄位，`filter ispresent(@transformationError)`以尋找所有失敗的轉換事件。

使用 Amazon OpenSearch Service 進行分析

CloudWatch Logs 與 整合， Amazon OpenSearch Service 可讓您建立自動策劃的儀表板，顯示 OpenSearch Service 從服務 AWS 提供的日誌衍生的關鍵指標。下列儀表板可供使用：

- Amazon VPC 流程日誌儀表板會擷取 Amazon VPC 的網路流程資料。它可協助您分析網路流量、偵測異常模式，以及監控資源使用情況。顯示的關鍵指標包括下列項目：
  - 這些流程的總流程和接受和拒絕
  - 一段時間內的流量模式
  - Sankey 圖表，說明來源和目的地 IPs之間的資料流程 （最佳發言者）
  - 依位元組和封包傳輸的熱門 IPs

  > **Note**
  >
  > 目前僅支援 VPC 第 2 版欄位格式。

- AWS WAF 日誌儀表板提供受監控之 Web 流量的洞見 AWS WAF。此儀表板可協助您識別來自特定區域或 IPs流量模式、封鎖請求和潛在威脅。顯示的關鍵指標包括下列項目：
  - 請求總數，包括「ALLOW」和「BLOCK」計數。
  - 隨時間的請求歷史記錄，顯示允許和封鎖的請求。
  - 依 Web ACL 名稱的請求明細、終止規則的封鎖請求，以及來源 IPs。
  - 請求原始伺服器的地理分佈。
  - 最高用戶端 IPs和依請求計數終止規則。
- CloudTrail 日誌儀表板提供使用 CloudTrail 日誌之 AWS 環境中 API 活動的概觀。它有助於監控 API 活動、稽核動作，以及識別潛在的安全或合規問題。顯示的關鍵指標包括下列項目：
  - 隨時間變化的總事件計數和事件歷史記錄
  - 依帳戶 IDs、類別和區域分類的事件明細。
  - 產生事件時涉及的熱門 APIs、服務和來源 IPs。
  - 產生事件的熱門使用者資料表，詳細說明使用者帳戶資訊和事件計數。
- AWS Network Firewall 儀表板可增強對網路流量的可見性，為安全監控和分析提供寶貴的洞見。此儀表板提供各種網路指標和模式的完整檢視，以快速識別潛在的安全問題並最佳化網路組態。顯示的關鍵指標包括下列項目：
  - 熱門發言者和通訊協定

- PrivateLink 端點的洞見
- 允許和封鎖的 TLS 伺服器名稱指示流量

這些精選儀表板中顯示的指標衍生自 Amazon OpenSearch Service 分析。

您必須先建立 IAM 角色並執行 CloudWatch Logs 的一次性整合，才能檢視這些儀表板 Amazon OpenSearch Service。此一次性整合會設定建立和轉譯儀表板所需的 Amazon OpenSearch Service 資源。您將需要為所使用的 OpenSearch 服務支付費用。如需詳細資訊，請參閱 Amazon CloudWatch 定價。

您只能為標準日誌類別中的日誌群組建立這些精心策劃的儀表板。

> **Important**
>
> 請勿將日誌轉換器用於任何您要為其建立付費日誌儀表板的日誌群組。轉換日誌事件會導致儀表板具有空白資料。

主題

步驟 1：建立與 OpenSearch Service 的整合

第一步是建立與 OpenSearch Service 的整合，您只需執行一次。建立整合會在您的帳戶中建立下列資源。

- 沒有高可用性OpenSearch Service 的時間序列集合。

  集合是一組 OpenSearch Service 索引，可一起運作以支援工作負載。
- 集合的兩個安全政策。一個定義加密類型，其具有客戶受管 AWS KMS 金鑰或服務擁有的金鑰。另一個政策定義網路存取，允許 OpenSearch Service 應用程式存取集合。如需詳細資訊，請參閱 Amazon OpenSearch Service 的靜態資料加密。

- OpenSearch Service 資料存取政策，定義誰可以存取集合中的資料。
- 將 CloudWatch Logs 定義為來源的 OpenSearch Service 直接查詢資料來源。
- 名稱為 的 OpenSearch Service 應用程式`aws-analytics`。應用程式將設定為允許建立工作區。如果名為 的應用程式`aws-analytics`已存在，則會將其更新為新增此集合做為資料來源。
- OpenSearch Service 工作區，將託管儀表板，並允許已獲得存取權限的每個人從工作區讀取。

主題

所需的許可

若要建立整合，您必須登入具有 CloudWatchOpenSearchDashboardsFullAccess 受管 IAM 政策或同等許可的帳戶，如下所示。您還必須擁有這些許可，才能刪除整合、建立、編輯和刪除儀表板，以及手動重新整理儀表板。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Sid": "CloudWatchOpenSearchDashboardsIntegration",
            "Effect": "Allow",
            "Action": [
                "logs:ListIntegrations",
                "logs:GetIntegration",
                "logs:DeleteIntegration",
                "logs:PutIntegration",
                "logs:DescribeLogGroups",
                "opensearch:ApplicationAccessAll",
                "iam:ListRoles",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLogsOpensearchReadAPIs",
            "Effect": "Allow",
```

```
            "Action": [
                "aoss:BatchGetCollection",
                "aoss:BatchGetLifecyclePolicy",
                "es:ListApplications"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsOpensearchCreateServiceLinkedAccess",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
opensearchservice.amazonaws.com/AWSServiceRoleForAmazonOpenSearchService",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "opensearchservice.amazonaws.com",
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsObservabilityCreateServiceLinkedAccess",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
observability.aoss.amazonaws.com/AWSServiceRoleForAmazonOpenSearchServerless",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "observability.aoss.amazonaws.com",
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsCollectionRequestAccess",
```

```
            "Effect": "Allow",
            "Action": [
                "aoss:CreateCollection"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:RequestTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "CloudWatchOpenSearchIntegration"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsApplicationRequestAccess",
            "Effect": "Allow",
            "Action": [
                "es:CreateApplication"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:RequestTag/OpenSearchIntegration": [
                        "Dashboards"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "OpenSearchIntegration"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsCollectionResourceAccess",
            "Effect": "Allow",
            "Action": [
                "aoss:DeleteCollection"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ]
                }
            }
        },
        {
            "Sid": "CloudWatchLogsApplicationResourceAccess",
            "Effect": "Allow",
            "Action": [
                "es:UpdateApplication",
                "es:GetApplication"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:ResourceTag/OpenSearchIntegration": [
                        "Dashboards"
                    ]
                }
            }
        },
        {
            "Sid": "CloudWatchLogsCollectionPolicyAccess",
            "Effect": "Allow",
            "Action": [
                "aoss:CreateSecurityPolicy",
                "aoss:CreateAccessPolicy",
                "aoss:DeleteAccessPolicy",
                "aoss:DeleteSecurityPolicy",
                "aoss:GetAccessPolicy",
                "aoss:GetSecurityPolicy"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aoss:collection": "cloudwatch-logs-*",
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                }
            }
        },
```

```
        {
            "Sid": "CloudWatchLogsAPIAccessAll",
            "Effect": "Allow",
            "Action": [
                "aoss:APIAccessAll"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aoss:collection": "cloudwatch-logs-*"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsIndexPolicyAccess",
            "Effect": "Allow",
            "Action": [
                "aoss:CreateAccessPolicy",
                "aoss:DeleteAccessPolicy",
                "aoss:GetAccessPolicy",
                "aoss:CreateLifecyclePolicy",
                "aoss:DeleteLifecyclePolicy"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aoss:index": "cloudwatch-logs-*",
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsDQSRequestQueryAccess",
            "Effect": "Allow",
            "Action": [
                "es:AddDirectQueryDataSource"
            ],
            "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:RequestTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ]
```

```
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "CloudWatchOpenSearchIntegration"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsStartDirectQueryAccess",
            "Effect": "Allow",
            "Action": [
                "opensearch:StartDirectQuery",
                "opensearch:GetDirectQuery"
            ],
            "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*"
        },
        {
            "Sid": "CloudWatchLogsDQSResourceQueryAccess",
            "Effect": "Allow",
            "Action": [
                "es:GetDirectQueryDataSource",
                "es:DeleteDirectQueryDataSource"
            ],
            "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ]
                }
            }
        },
        {
            "Sid": "CloudWatchLogsPassRoleAccess",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService":
"directquery.opensearchservice.amazonaws.com",
                    "aws:CalledViaFirst": "logs.amazonaws.com"
```

```
                }
            }
        },
        {
            "Sid": "CloudWatchLogsAossTagsAccess",
            "Effect": "Allow",
            "Action": [
                "aoss:TagResource"
            ],
            "Resource": "arn:aws:aoss:*:*:collection/*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "logs.amazonaws.com",
                    "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "CloudWatchOpenSearchIntegration"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsEsApplicationTagsAccess",
            "Effect": "Allow",
            "Action": [
                "es:AddTags"
            ],
            "Resource": "arn:aws:opensearch:*:*:application/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/OpenSearchIntegration": [
                        "Dashboards"
                    ],
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "OpenSearchIntegration"
                }
            }
        },
        {
            "Sid": "CloudWatchLogsEsDataSourceTagsAccess",
            "Effect": "Allow",
```

```
            "Action": [
                "es:AddTags"
            ],
            "Resource": "arn:aws:opensearch:*:*:datasource/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                        "Dashboards"
                    ],
                    "aws:CalledViaFirst": "logs.amazonaws.com"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "CloudWatchOpenSearchIntegration"
                }
            }
        }
    ]
}
```

建立整合

使用這些步驟來建立整合。

將 CloudWatch Logs 與 整合 Amazon OpenSearch Service

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇 Logs Insights，然後選擇使用 OpenSearch 分析索引標籤。
3. 選擇建立整合。
4. 對於整合名稱，輸入整合的名稱。
5. （選用）若要加密寫入 OpenSearch Service Serverless 的資料，請輸入您要在 KMS AWS KMS 金鑰 ARN 中使用的金鑰 ARN。 如需詳細資訊，請參閱《Amazon OpenSearch Service 開發人員指南》中的靜態加密。
6. 對於資料保留，輸入您希望保留 OpenSearch Service 資料索引的時間量。這也會定義您可以在儀表板中檢視資料的最長期間。選擇較長的資料保留期會產生額外的搜尋和索引成本。如需詳細資訊，請參閱 OpenSearch Service Serverless 定價。

   最長保留期間為 30 天。

   資料保留長度也會用來建立 OpenSearch Service 收集生命週期政策。

7. 對於要寫入 OpenSearch 集合的 IAM 角色，建立新的 IAM 角色或選取要用來寫入 OpenSearch Service 集合的現有 IAM 角色。

   建立新角色是最簡單的方法，該角色將以必要的許可建立。

   > **Note**
   >
   > 如果您建立角色，它將具有從帳戶中的所有日誌群組讀取的許可。

   如果您想要選取現有的角色，它應該具有 中列出的許可整合所需的許可。或者，您可以選擇使用現有角色，然後在驗證所選角色的存取許可區段中選擇建立角色。如此一來，您就可以使用 中列出的許可整合所需的許可做為範本並進行修改。例如，如果您想要指定更精細的日誌群組控制。

8. 對於可以檢視儀表板的 IAM 角色和使用者，您可以選取如何將存取權授予 IAM 角色和 IAM 使用者，以便取得日誌儀表板存取權：

   - 若要限制只有部分使用者的儀表板存取，請選擇選取 IAM 角色和可以檢視儀表板的使用者，然後在文字方塊中搜尋並選取您要授予存取權的 IAM 角色和 IAM 使用者。
   - 若要將儀表板存取權授予所有使用者，請選擇允許此帳戶中的所有角色和使用者檢視儀表板。

   > **Important**
   >
   > 選取角色或使用者，或選擇所有使用者，只會將他們新增至存取儲存儀表板資料的 OpenSearch Service 集合所需的資料存取政策。 OpenSearch 若要讓他們能夠檢視付費日誌儀表板，您還必須授予這些角色和使用者 CloudWatchOpenSearchDashboardAccess受管 IAM 政策。

9. 選擇建立整合

   建立整合需要幾分鐘的時間。

步驟 2：建立付費日誌儀表板

建立整合之後，您可以建立儀表板。儀表板可用於 Amazon VPC 流程日誌、CloudTrail 日誌和 AWS WAF 日誌。

使用 OpenSearch Service 衍生的指標建立付費日誌儀表板

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。

2. 在左側導覽窗格中，選擇 Logs Insights，然後選擇使用 OpenSearch 分析索引標籤。
3. 選擇 Create dashboard (建立儀表板)。
4. 選擇要建立儀表板的日誌類型 AWS WAF、Amazon VPC 流程日誌、CloudTrail 或 AWS Network Firewall。
5. 輸入儀表板的名稱，然後選擇性地輸入描述。
6. 對於資料同步頻率，輸入您希望 OpenSearch Service 查詢 CloudWatch 的頻率，以便使用新資料同步和更新 OpenSearch Service 中建立的指標和索引。OpenSearch Service 會在日誌上建立指標和索引，以轉譯儀表板。

   選擇較短的時間可讓資料保持在最新狀態，並產生更高的成本。
7. 選取要從中收集此儀表板資料的日誌群組。請務必選取符合您建立之儀表板類型的日誌群組。

   您可以在進行這些選擇時，使用瀏覽日誌群組按鈕和檢視所選日誌群組的日誌範例選項，以確保您取得所需的日誌群組。
8. 選擇 Create dashboard (建立儀表板)。

   一開始，儀表板不會顯示任何資料。幾分鐘後，資料會出現在儀表板中。資料第一次出現時，將會是最近 15 分鐘的日誌項目。

檢視、編輯或刪除結束的日誌儀表板

在 CloudWatch Logs 或 OpenSearch Service 中檢視付費日誌儀表板

若要檢視儀表板，您必須登入具有 CloudWatchOpenSearchDashboardAccess IAM 政策的 IAM 主體。

檢視付費日誌儀表板

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇 Logs Insights，然後選擇使用 OpenSearch 分析索引標籤。
3. 在 OpenSearch 儀表板方塊中選擇儀表板。
4. （選用）在右上角，選擇在 OpenSearch 中檢視。

   OpenSearch Service 主控台隨即開啟，您會在那裡看到相同的儀表板。在 OpenSearch Service 主控台中，您可以變更儀表板及其小工具，而且當您在 CloudWatch Logs 中檢視儀表板時，也會顯示這些變更。

授予儀表板檢視其他 IAM 角色或 IAM 使用者的存取權

若要在建立整合之後授予其他 IAM 主體的存取權，請執行下列步驟。

授予附加日誌儀表板存取權給其他 IAM 角色或使用者

1. 編輯集合的資料存取政策，以新增這些角色或使用者。如需詳細資訊，請參閱《OpenSearch Service 開發人員指南》中的 Amazon OpenSearch Service Serverless 的資料存取控制。OpenSearch
2. 將 CloudWatchOpenSearchDashboardAccess 授予這些使用者。如需此政策內容的詳細資訊，請參閱 CloudWatchOpenSearchDashboardAccess。

編輯儀表板組態

您可以編輯現有付費日誌儀表板的名稱、描述和同步頻率。

編輯已發佈的日誌儀表板

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇 Logs Insights，然後選擇使用 OpenSearch 分析索引標籤。
3. 在 OpenSearch 儀表板方塊中選擇儀表板。
4. 選擇動作、變更儀表板詳細資訊。
5. 進行變更，然後選擇確認變更。

刪除已發佈的日誌儀表板

您可以刪除已結束的日誌儀表板。如果您這樣做，在 OpenSearch Service 集合中建立的儀表板、指標和索引都會遭到刪除。

> **Note**
>
> 刪除結束的日誌儀表板後，請等待至少六小時，然後再嘗試重新建立相同的儀表板。如果您不等待，重新建立的儀表板將無法正常運作。

刪除已發佈的日誌儀表板

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側導覽窗格中，選擇 Logs Insights，然後選擇使用 OpenSearch 分析索引標籤。
3. 在 OpenSearch 儀表板方塊中選擇儀表板。
4. 選擇 動作、刪除。
5. 輸入 來確認您的決策**delete**，然後選擇刪除。

刪除所有與 OpenSearch Service 的付費日誌儀表板整合

您可以刪除整個 OpenSearch 整合。如果您這麼做，則會刪除所有付費日誌儀表板和其中顯示的資料。

> **Important**
>
> 為了避免持續成本，強烈建議您在刪除整合之前手動刪除下列資源。刪除整合不會自動刪除這些資源，而且在刪除整合之後，您將無法存取這些資源來刪除這些資源。若要尋找要刪除的資源名稱，請參閱下列程序。
>
> - 資料來源
> - 集合
> - 資料存取政策
> - 加密政策
> - 網路政策
> - 生命週期政策

刪除與 OpenSearch Service 的整個付費日誌儀表板整合

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在左側的導覽窗格中，選擇設定。
3. 選擇 Logs (日誌) 索引標籤。
4. 在 OpenSearch 整合區段中，選擇刪除整合。

   下一個畫面會顯示您在刪除整合之前應刪除的 OpenSearch Service 資源名稱。

5. 輸入 來確認您的決策**delete**，然後選擇刪除整合。

使用者的 IAM 政策

CloudWatch Logs 已建立兩個 IAM 政策：CloudWatchOpenSearchDashboardsFullAccess 和 CloudWatchOpenSearchDashboardAccess。下表列出每個政策啟用的動作。

Action	IAM 政策	所需的其他許可
建立整合	CloudWatchOpenSearchDashboardsFullAccess	
刪除整合	CloudWatchOpenSearchDashboardsFullAccess	
建立儀表板	CloudWatchOpenSearchDashboardsFullAccess	
編輯儀表板	CloudWatchOpenSearchDashboardsFullAccess	
刪除儀表板	CloudWatchOpenSearchDashboardsFullAccess	
立即使用同步重新整理儀表板	CloudWatchOpenSearchDashboardsFullAccess	
在設定中檢視整合	CloudWatchOpenSearchDashboardAccess 或 CloudWatchOpenSearchDashboardsFullAccess	
檢視儀表板	CloudWatchOpenSearchDashboardAccess 或 CloudWatchOpenSearchDashboardsFullAccess	當您建立整合時指定角色或使用者，或編輯集合的資料存取政策，以新增這些角色或使用者。如需詳細資訊，請參閱《OpenSearch Service 開發人員指南》中的

Action	IAM 政策	所需的其他許可
		Amazon OpenSearch Service Serverless 的資料存取控制。OpenSearch
在 OpenSearch Service 主控台中檢視儀表板	CloudWatchOpenSearchDashboardAccess 或 CloudWatchOpenSearchDashboardsFullAccess	當您建立整合時指定角色或使用者，或編輯集合的資料存取政策，以新增這些角色或使用者。如需詳細資訊，請參閱《OpenSearch Service 開發人員指南》中的 Amazon OpenSearch Service Serverless 的資料存取控制。OpenSearch

整合所需的許可

如果您為要使用的整合建立 IAM 角色，而不是允許 CloudWatch Logs 建立角色，則必須包含下列許可和信任政策。如需如何建立 IAM 角色的詳細資訊，請參閱建立角色以將許可委派給 AWS 服務。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "CloudWatchLogsAccess",
      "Effect": "Allow",
      "Action": [
        "logs:StartQuery",
        "logs:GetLogGroupFields",
        "logs:GetQueryResults"
      ],
      "Resource": [
        "*"
      ]
    },
    {
```

```
      "Sid": "CloudWatchLogsDescribeLogGroupsAccess",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    },
    {
        "Sid": "AmazonOpenSearchCollectionAccess",
        "Effect": "Allow",
        "Action": [
            "aoss:APIAccessAll"
        ],
        "Resource": "*",
        "Condition": {
            "StringLike": {
                "aoss:collection": "cloudwatch-logs-*"
            }
        }
    }
  ]
}
```

Note

先前的角色授予從 帳戶中所有日誌群組讀取的存取權，讓您能夠為任何日誌帳戶建立儀表板，包括跨帳戶日誌群組。如果您想要限制對特定日誌群組的存取，並僅為這些日誌群組建立儀表板，您可以將該政策中的第一個陳述式更新為下列項目：

```
{
      "Sid": "CloudWatchLogsAccess",
      "Effect": "Allow",
      "Action": [
        "logs:StartQuery",
        "logs:GetLogGroupFields",
        "logs:GetQueryResults"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:123456789012:log-group:myLogGroup:*",
        "arn:aws:logs:us-east-1:123456789012:log-group:myLogGroup"
      ]
```

```
}
```

使用 S3 Tables 整合存取日誌

S3 Tables 與 CloudWatch 整合可讓您使用分析引擎存取擷取至 CloudWatch 的日誌資料，例如 Amazon Athena、Amazon Redshift，以及支援連線至 Apache Iceberg 相容存放區的第三方工具。此整合可讓您使用偏好的工具執行全面的日誌分析，並將 CloudWatch Logs 中的資料與非 CloudWatch 資料建立關聯。

了解 S3 資料表整合

Amazon S3 Tables Integration 是一種全受管解決方案，可讓您在 CloudWatch Logs 中的日誌做為受管 Amazon S3 資料表使用。透過此整合，除了 CloudWatch Logs 功能之外，您還可以獲得更靈活的分析日誌方式。

整合的運作方式是建立受管 Amazon S3 資料表儲存貯體 (`aws-cloudwatch`)，並根據資料來源名稱和類型將特定日誌來源與 Amazon S3 資料表建立關聯 （ 可從 CloudWatch Logs 主控台中的日誌管理 > 資料來源索引標籤進行管理 ）。建立關聯後，CloudWatch Logs 資料即可使用 Apache Iceberg 格式透過 Amazon S3 Tables 存取。此格式為各種分析引擎提供標準化的方式來有效率地查詢資料。

核心元件

資料來源關聯

根據資料來源和類型條件，將特定 CloudWatch Logs 來源連結至 S3 Tables 整合的程序。

Apache Iceberg 資料表

S3 Tables 所使用的基礎資料表格式，可提供結構化資料儲存，並與多個分析引擎相容。

S3 資料表的資料流程

了解 CloudWatch Logs 和 S3 Tables 之間的資料流程如何協助您規劃整合並有效管理日誌資料。

當您建立關聯時，CloudWatch Logs 會自動將符合相關聯資料來源名稱和類型的新日誌事件傳送至 CloudWatch 受管 S3 資料表儲存貯體。您可以在該資料來源的對應資料表下的日誌命名空間中找到這些事件。整合只會處理在您建立關聯之後新增的日誌事件，而且不會在建立關聯之前從 回填日誌。

S3 資料表儲存貯體中的資料保留符合為日誌群組設定的保留政策。例如，如果您將日誌群組設定為 1 天保留，CloudWatch Logs 會在一天後從 CloudWatch Logs 和 S3 資料表中移除資料。當您刪除日誌群組或日誌串流時，CloudWatch Logs 也會從 S3 資料表儲存貯體中移除資料。

何時使用 S3 資料表整合

考慮使用 S3 Tables 整合將日誌資料與其他外部或非 CloudWatch 資料建立關聯，或當您偏好使用 Amazon Athena 等其他分析工具對 CloudWatch Logs 資料執行分析時。當您需要的功能超出 CloudWatch Logs 中可用的功能時，請使用此整合。此整合在以下情況下特別有用：

- 您需要跨大量日誌資料執行複雜的類似 SQL 的查詢
- 您想要將日誌分析與現有的分析工作流程和工具整合
- 您需要涵蓋多個資料來源的全方位日誌分析功能

除了現有的 CloudWatch 擷取和儲存定價之外，透過此整合建立的 S3 資料表無需支付額外的儲存或資料表維護費用。

先決條件

在實作整合之前，請確定您有下列項目：

- 現有的 CloudWatch Logs 資料
- CloudWatch Logs 和 S3 Tables 之間跨服務存取的適當 IAM 許可，如下節所述

IAM 許可

若要將 CloudWatch Logs 與 S3 Tables 整合，您需要為兩個不同的實體設定 IAM 許可：設定整合的使用者或角色，以及 CloudWatch Logs 擔任以將資料寫入 S3 Tables 的服務角色。

針對建立整合的角色或使用者

設定整合的使用者或角色需要下列許可：

- `observabilityadmin:CreateS3TableIntegration` 建立整合並`logs:AssociateSourceToS3TableIntegration`新增來源
- `s3tables:CreateTableBucket`、 `s3tables:PutTableBucketEncryption`和 `s3tables:PutTableBucketPolicy` 來設定 S3 資料表儲存貯體

對於服務角色

將下列 IAM 政策連接至 CloudWatch Logs 用來將資料寫入資料表儲存貯體的 IAM 服務角色。此政策授予寫入資料表的許可。將 *aws-region*、*123456789012* 和 *log-group-name* 取代為您的 AWS 區域、帳戶 ID 和日誌群組名稱。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:integrateWithS3Table"
            ],
            "Resource": ["arn:aws:logs:aws-region:123456789012:log-group:log-group-
name"],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

將下列信任政策連接至 CloudWatch Logs 將擔任的 IAM 服務角色，以將日誌資料寫入 S3 Tables。您可以在整合設定期間建立或選取此角色。這些條件會限制角色，以便 CloudWatch Logs 只能為指定的帳戶和日誌群組擔任該角色。將 *aws-region*、*123456789012* 和 *log-group-name* 取代為您的 AWS 區域、帳戶 ID 和日誌群組名稱。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
```

```
                },
                "ArnLike": {
                    "aws:SourceArn": ["arn:aws:logs:aws-region:123456789012:log-
group:log-group-name"]
                }
            }
        }
    ]
}
```

KMS 金鑰政策（適用於加密資料）

如果您使用客戶受管金鑰來加密日誌資料，則必須授予 CloudWatch 服務主體和 S3 Tables 維護服務主體存取金鑰的權限。將下列陳述式新增至您的 KMS 金鑰政策。將預留位置值取代為您的 AWS 帳戶 ID、區域、KMS 金鑰 ID 和 S3 資料表或資料表儲存貯體 ARN。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnableSystemTablesKeyUsage",
            "Effect": "Allow",
            "Principal": {
                "Service": "systemtables.cloudwatch.amazonaws.com"
            },
            "Action": [
                "kms:DescribeKey",
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:aws-region:123456789012:key/key-id",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        },
        {
            "Sid": "EnableKeyUsage",
            "Effect": "Allow",
            "Principal": {
                "Service": "maintenance.s3tables.amazonaws.com"
            },
```

```
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:aws-region:123456789012:key/key-id",
            "Condition": {
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn": "<table-or-table-bucket-arn>/*"
                }
            }
        }
    ]
}
```

開始使用

若要開始使用 S3 Tables Integration，您需要設定 CloudWatch Logs 和 S3 Tables 之間的整合。此程序涉及設定資料來源關聯和設定適當的 IAM 許可。

建立 S3 資料表整合

1. 開啟位於 https：//https://console.aws.amazon.com/cloudwatch/ 的 CloudWatch Logs 主控台。
2. 選擇設定、全域、建立 S3 資料表整合。
3. 自訂如何在 S3 Tables 中加密日誌，以及 CloudWatch Logs 將用來將日誌寫入 S3 Tables 的角色。
4. 選擇建立 S3 資料表整合。

將來源與 S3 資料表整合建立關聯

1. 在 https：//https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch Logs 主控台。
2. 選擇設定、全域、管理 S3 資料表整合。
3. 選擇關聯資料來源。
4. 選取您要啟用整合的資料來源名稱和資料來源類型。
5. 選擇關聯資料來源。

從日誌管理頁面將來源與 S3 資料表整合建立關聯

1. 在 https：//https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch Logs 主控台。

2. 在導覽窗格中選擇日誌管理。
3. 選取資料來源索引標籤。
4. 選擇您要整合的資料來源名稱和資料來源類型。
5. 選擇資料來源動作。
6. 選取與 S3 Tables Integration 建立關聯。
7. 檢閱資料來源，然後選擇關聯資料來源。

您必須先執行下列 3 個步驟，才能使用資料：

1. 將 Amazon S3 Tables 與 AWS 分析服務整合 - 使用 Amazon S3 主控台
2. 設定 Lake Formation 許可
3. 使用分析工具連線

將 Amazon S3 Tables 與 AWS 分析服務整合 - 使用 Amazon S3 主控台 (連結)

使用 S3 主控台啟用 S3 Tables 整合

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 在左側導覽窗格中，選擇資料表儲存貯體。
3. 按一下頂端的啟用整合。
4. 您第一次在任何區域中整合資料表儲存貯體時，Amazon Amazon S3 會代表您建立新的 IAM 服務角色。此角色允許 Lake Formation 存取您帳戶中的所有資料表儲存貯體，並聯合存取 Glue Data Catalog AWS 中的資料表。

設定 Lake Formation 許可

雖然 CloudWatch Logs 具有寫入資料表的許可 （在先前的步驟中設定），但使用者和分析角色不會自動擁有讀取資料的許可。您必須使用 AWS Lake Formation 明確授予存取權。您必須為每個要提供資料表存取權的 IAM 主體執行此操作。

將查詢存取權授予使用者或角色

您必須將 SELECT 和 DESCRIBE 許可授予將在 Athena 或 Redshift 中執行查詢的 IAM 主體 （使用者或角色）。

1. 開啟 AWS Lake Formation 主控台。
2. 在導覽窗格中的許可下，選擇資料湖許可。
3. 選擇 Grant (授予)。
4. 委託人：選取需要存取的 IAM 使用者或角色 （例如，您的資料分析師或您目前正在使用的管理員角色）。
5. LF 標籤或目錄資源：選取具名資料目錄資源。
6. 資料庫和資料表：
   - 選取 CloudWatch 整合所建立的 S3 Table 儲存貯體 (`aws-cloudwatch`)。
   - 選取與您的資料來源相關聯的特定資料表 （選用）。
7. 資料表許可：選取選取和描述。
8. 選擇 Grant (授予)。

> **Note**
>
> 如果您在 Athena 中查詢日誌時遇到「存取遭拒」錯誤，請確定執行查詢的使用者具有上述定義之 Athena 和 Lake Formation 許可的適當 IAM 許可。

如需 Lake Formation 許可的詳細資訊，請參閱 https：//https://docs.aws.amazon.com/lake-formation/latest/dg/granting-catalog-permissions.html。

使用分析工具連線

授予許可後，您可以設定偏好的分析服務來查詢 S3 資料表。S3 Tables 使用 Apache Iceberg 格式，Amazon Athena、Amazon Redshift 和 Amazon EMR 原生支援此格式。

在 Amazon Athena 中查詢日誌資料

Amazon Athena 透過 Amazon S3 Tables 目錄與 Amazon S3 Tables 互動。

設定 Athena 以查詢您的日誌資料

1. 前往 https://console.aws.amazon.com/athena/ 開啟 Amazon Athena 主控台。
2. 在查詢編輯器中，從資料來源下拉式清單中選取 Amazon S3 Tables 目錄。
3. 如果您沒有看到目錄，請確定您已完成上述特定使用者角色的 Lake Formation 許可步驟。

4. 選取目錄後，您的日誌資料表將顯示在資料庫清單中。您現在可以針對日誌資料執行標準 SQL 查詢。

查詢範例： `SELECT * FROM "amazon_vpc__flow" LIMIT 100;`

請前往 https：//https://docs.aws.amazon.com/AmazonS3/latest/userguide/s3-tables-integrating-aws.html 進一步了解如何將 Analytics 服務與 S3 Tables 連線。

使用篩選條件從日誌事件建立指標

您可以建立一個或多個指標篩選條件，以搜尋並篩選進入 CloudWatch Logs 的日誌資料。指標篩選條件可定義要在傳送至 CloudWatch Logs 的日誌資料中尋找的詞彙和模式。CloudWatch Logs 使用這些指標篩選條件將日誌資料轉換為數值 CloudWatch 指標，供您繪製圖形或設定警示。

當您從記錄篩選條件建立指標時，您也可以選擇指派指標的維度和單位。如果您指定單位，請務必在建立篩選條件時指定正確的單位。後來再變更篩選條件的單位沒有作用。

如果您已設定 AWS Organizations 並正在使用成員帳戶，您可以使用日誌集中，從來源帳戶收集日誌資料到中央監控帳戶。

使用集中式日誌群組時，您可以在建立指標篩選條件時使用這些系統欄位維度。

- `@aws.account` - 此維度代表日誌事件來源 AWS 的帳戶 ID。
- `@aws.region` - 此維度代表產生日誌事件 AWS 的區域。

這些維度有助於識別日誌資料來源，以便更精細地篩選和分析衍生自集中式日誌的指標。如需詳細資訊，請參閱跨帳戶跨區域日誌集中化。

如果具有訂閱的日誌群組使用日誌轉換，則篩選條件模式會套用至日誌事件的轉換版本。如需詳細資訊，請參閱在擷取期間轉換日誌。

> **Note**
>
> 僅標準日誌類別中的日誌群組支援指標篩選條件。如需日誌類別的詳細資訊，請參閱 日誌類別。

當檢視這些指標或設定警示時，您可以使用任何類型的 CloudWatch 統計數字，包括百分位數統計數字。

> **Note**
>
> 只有在指標值全都不是負數時，才會支援百分位數統計資料。如果您設定指標篩選條件，使其可以回報負數，百分位數統計資料在擁有負數的值時將無法用於該指標。如需詳細資訊，請參閱百分位數。

篩選條件不追溯篩選條件資料。篩選條件只針對篩選條件建立後發生的事件發佈指標資料點。測試篩選條件模式時，篩選條件結果預覽最多會顯示前 50 個相符的日誌行，以供驗證之用。如果篩選結果上的時間戳記早於指標建立時間，則不會顯示日誌。

目錄

概念

每個指標篩選條件是由下列關鍵元素組成：

預設值

值會在日誌被擷取但沒有發現相符日誌的時間段內，回報至指標篩選條件。將此設定為 0，可確保每個時間段都會回報資料，以免某些時間段沒有相符的資料，而產生起伏不定的指標。不過，如果在 1 分鐘的時間段內沒有擷取任何日誌事件，則不會回報任何值。

如果您將維度指派給由指標篩選條件建立的指標，則無法為該指標指派預設值。

維度

維度是進一步定義指標的鍵值組。您可以將維度指派給從指標篩選條件建立的指標。由於維度是指標唯一識別符的一部分，每當您從日誌擷取唯一名稱/值組時，就是在建立該指標的新變化。

篩選條件模式

象徵性描述 CloudWatch Logs 應如何解譯每個日誌事件中的資料。例如，日誌項目可能包含時間戳記、IP 地址、字串，以此類推。您可以使用模式以指定要在日誌檔中尋找的項目。

指標名稱

受監控日誌資訊應發佈至其中的 CloudWatch 指標名稱。例如，您可能發佈到名為 ErrorCount 的指標。

指標命名空間

新 CloudWatch 指標的目的地命名空間。

指標值

每次找到相符日誌時要發佈到指標的數值。例如，如果您是計數特定詞彙 (像是 "Error") 的出現次數，每個出現次數的值將為 "1"。如果您是計數傳出的位元組，您可以透過日誌事件中找到的實際位元組數來遞增計數。

指標篩選條件的篩選條件模式語法

> **Note**
>
> 指標篩選條件與 CloudWatch Logs Insights 查詢有何不同
> 指標篩選條件與 CloudWatch Logs Insights 查詢的不同之處在於，每次找到相符的日誌時，都會將指定的數值新增至指標篩選條件。如需詳細資訊，請參閱配置指標篩選條件的指標值。
> 如需有關如何使用 Amazon CloudWatch Logs Insights 查詢語言查詢日誌群組的資訊，請參閱 CloudWatch Logs Insights 語言查詢語法。
> 一般篩選條件模式範例
> 如需適用於指標篩選條件以及訂閱篩選條件和篩選條件日誌事件的一般篩選條件模式語法，請參閱適用於指標篩選條件、訂閱篩選條件和篩選條件日誌事件的篩選條件模式語法，其中包括下列範例：
>
> - 支援的規則運算式 (regex) 語法
> - 在非結構化日誌事件中比對詞彙
> - 在 JSON 日誌事件中比對詞彙
> - 比對以空格分隔的日誌事件中的詞彙

指標篩選條件可讓您搜尋和篩選進入 CloudWatch Logs 的日誌資料、從篩選的日誌資料中擷取指標觀測項目，以及將資料點轉換為 CloudWatch Logs 指標。定義要在傳送至 CloudWatch Logs 的日誌資料中尋找的詞彙和模式。指標篩選條件指派給日誌群組，並且指派給日誌群組的所有篩選條件都會套用於其日誌串流。

當指標篩選條件與某個詞彙相符時，便會以指定的數值增加指標的計數。例如：您可以建立指標篩選條件來搜尋與計算日誌事件中單字 ERROR (錯誤) 的出現次數。

您可以對指標指派度量單位和維度。例如：如果您建立了一個指標篩選條件，以計算單字 ERROR (錯誤) 在日誌事件中出現的次數，則可以指定一個名為 `ErrorCode` 的維度，以顯示包含 ERROR (錯誤) 這個單字的日誌事件總數，並根據回報的錯誤代碼篩選資料。

> **Tip**
>
> 指派度量單位給指標時，務必指定正確的單位。如果隨後再更改單位，則更改可能無法生效。如需 CloudWatch 支援的單位完整清單，請參閱 Amazon CloudWatch API 參考 中的 MetricDatum。

主題

配置指標篩選條件的指標值

建立指標篩選條件時，您可以定義篩選條件模式，並指定指標的值和預設值。您可以將指標值設置為數字、名稱標識符或數字識別碼。如果您未指定預設值，指標篩選條件找不到相符項目時，CloudWatch 將不會回報資料。我們建議您指定預設值，即使該值為 0。設定預設值有助於 CloudWatch 更準確地回報資料，防止 CloudWatch 彙總起伏不定的指標。CloudWatch 每分鐘彙總並報告指標值。

指標篩選條件在日誌事件中找到相符項目時，會根據指標值增加指標的計數。如果指標篩選條件未找到相符項目，CloudWatch 會回報指標的預設值。例如：假設有一個日誌群組，每分鐘發佈兩個記錄且指標值為 1，預設值為 0。如果指標篩選條件第一分鐘內，在兩個日誌記錄中找到相符項目，則該分鐘的指標值為 2。如果指標篩選條件第二分鐘內，未在任何一筆記錄中找到相符項目，則該分鐘的預設值為 0。如果將維度分配給指標篩選條件產生的指標，則無法為這些指標指定預設值。

您也可以設置指標篩選條件，使用從日誌事件中擷取的值 (而非靜態值) 來增加指標。如需詳細資訊，請參閱使用日誌事件中的值來增加指標的值。

從 JSON 或空格分隔日誌事件的值中將維度連同指標一起發佈

您可以使用 CloudWatch 主控台或 AWS CLI 建立指標篩選條件，以使用 JSON 和空格分隔日誌事件產生的指標來發佈維度。維度是名稱/數值配對，僅適用於 JSON 和空格分隔的篩選條件模式。您可以建立最多包含三個維度的 JSON 和空格分隔的指標篩選條件。如需維度以及如何將維度指派給指標的詳細資訊，請參閱下列各節：

- 《Amazon CloudWatch 使用者指南》中的維度
- 《Amazon CloudWatch Logs 使用者指南》中的範例：從 Apache 日誌中擷取欄位並指派維度

Important

維度包含與自訂指標相同收費的值。為了避免費用出乎意料，請勿將高基數欄位指定為維度，例如 `IPAddress` 或 `requestID`。
從日誌事件擷取的指標會以自訂指標收費。為了避免費用意外過高，如果指標篩選條件針對您已指定的維度，在一定時間內產生 1000 個不同的名稱/值組，Amazon 可能會停用該指標篩選條件。
您可以建立帳單警示，通知您預估的費用。如需詳細資訊，請參閱建立帳單警示來監控預估的 AWS 費用。

從 JSON 日誌事件將維度連同指標一起發佈

下列範例程式碼片段描述如何在 JSON 指標篩選條件中指定維度。

Example: JSON log event

```
{
  "eventType": "UpdateTrail",
  "sourceIPAddress": "111.111.111.111",
  "arrayKey": [
        "value",
        "another value"
  ],
  "objectList": [
       {"name": "a",
         "id": 1
       },
       {"name": "b",
         "id": 2
       }
  ]

}
```

Note

如果要使用 JSON 日誌事件範例測試指標篩選條件範例，則必須在單行中輸入JSON 日誌範例。

Example: Metric filter

每當 JSON 日誌事件包含屬性 eventType 和 "sourceIPAddress" 時，指標篩選條件會增加指標。

```
{ $.eventType = "*" && $.sourceIPAddress != 123.123.* }
```

當您建立 JSON 指標篩選條件時，您可以將指標篩選條件中的任何屬性指定為維度。例如：若要設定 eventType 作為維度，請使用下列內容：

```
"eventType" : $.eventType
```

指標範例包含一個名為 "eventType" 的維度，其維度值在日誌事件範例中為 "UpdateTrail"。

從空格分隔日誌事件將維度連同指標一起發佈

下列範例程式碼片段，描述如何在空格分隔的指標篩選條件中指定維度。

Example: Space-delimited log event

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404
 1534
```

Example: Metric filter

```
[ip, server, username, timestamp, request, status_code, bytes > 1000]
```

當空格分隔的日誌事件包含篩選條件中指定的任何欄位時，該指標篩選條件會增加指標。例如：指標篩選條件在空格分隔的日誌事件範例中查找下列欄位和值。

```
{
   "$bytes": "1534",
   "$status_code": "404",

   "$request": "GET /index.html HTTP/1.0",
   "$timestamp": "10/Oct/2000:13:25:15 -0700",
   "$username": "frank",
   "$server": "Prod",
   "$ip": "127.0.0.1"
}
```

當您建立空格分隔的指標篩選條件時，您可以將指標篩選條件中的任何欄位指定為維度。例如：若要設定 server 作為維度，請使用下列內容：

```
"server" : $server
```

指標篩選條件範例中有一個名為 server 的維度，其維度值在日誌事件範例中為 "Prod"。

Example: Match terms with AND (&&) and OR (||)

您可以使用邏輯運算子邏輯與 ("&&") 和邏輯或 ("||") 建立包含條件的空格分隔的指標篩條件。下列指標篩選條件會傳回第一個單字為 ERROR 或 WARN 超級字串的日誌事件。

```
[w1=ERROR || w1=%WARN%, w2]
```

使用日誌事件中的值來增加指標的值

您可以建立指標篩選條件來發佈日誌事件中找到的數值。本節中的過程中使用下列範例指標篩選條件，來示範如何將 JSON 日誌事件中的數值發佈到指標。

```
{ $.latency = * } metricValue: $.latency
```

建立在日誌事件中發佈的值的指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Log groups (日誌群組)。

3. 選擇或建立日誌群組。

   如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的在 CloudWatch Logs 中建立日誌群組。

4. 選擇 Actions (動作)，然後選擇 Create metric filter (建立指標篩選條件)。
5. 針對 Filter Pattern (篩選條件模式)，輸入 **`{ $.latency = * }`**，然後選擇 Next (下一步)。
6. 針對 Metric Name (指標名稱)，輸入 myMetric。
7. 針對 Metric Value (指標值)，輸入 **`$.latency`**。
8. (選用) 針對 Default Value (預設值)，輸入 0，然後選擇 Next (下一步)。

   我們建議您指定預設值，即使該值為 0。設定預設值有助於 CloudWatch 更準確地回報資料，防止 CloudWatch 彙總起伏不定的指標。CloudWatch 每分鐘彙總並報告指標值。

9. 選擇 Create metric filter (建立指標篩選條件)。

指標篩選條件的範例與 JSON 日誌事件範例中的詞彙 `"latency"` 相符，並將數值 50 發佈到指標 myMetric。

```
{
"latency": 50,
"requestType": "GET"
}
```

建立指標篩選條件

下列程序和範例示範如何建立指標篩選條件。

範例

- 建立日誌群組的指標篩選條件
- 範例：計算日誌事件數量
- 範例：計算詞彙的出現次數
- 範例：Count HTTP 404 代碼
- 範例：Count HTTP 4xx 代碼
- 範例：從 Apache 日誌擷取欄位並指派維度

建立日誌群組的指標篩選條件

若要建立日誌群組的指標篩選條件，請遵循下列步驟。在有一些資料點之前，是看不到該指標的。

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Log groups (日誌群組)。
3. 選擇日誌群組的名稱。
4. 選擇 `Actions`，然後選擇 Create metric filter (建立指標篩選條件)。
5. 針對 Filter pattern (篩選條件模式)，輸入篩選條件模式。如需詳細資訊，請參閱用於指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法。
6. （選用）如果您使用集中式日誌群組，您可以在篩選條件選擇條件下，根據來源帳戶 (`@aws.account`)、來源區域 (`@aws.region`) 或兩者條件來指定篩選條件。
7. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件來測試模式。每個日誌事件必須在一行中格式化。分行符號用於分隔 Log event messages (日誌事件訊息) 方塊中的日誌事件。
8. 選擇 Next (下一步)，然後輸入指標篩選條件的名稱。
9. 在 Metric details (指標詳細資訊) 下的 Metric namespace (指標命名空間) 中，輸入將發佈指標的 CloudWatch 命名空間名稱。如果命名空間不存在，請務必選取 Create new (新建)。
10. 針對 Metric name (指標名稱)，輸入新指標的名稱。
11. 針對 Metric value (指標值)，如果指標篩選條件計算篩選條件中的關鍵字出現次數，請輸入 1。如此會針對包含其中一個關鍵字的每個日誌事件，將指標遞增 1。

    或者輸入字符，例如 **`$size`**。如此會針對包含 `size` 欄位的每個日誌事件，以 `size` 欄位中的數值遞增指標。
12. (選用) 針對 Unit (單位)，選取要指派給指標的單位。如果您未指定單位，單位會設為 None。
13. (選用) 最多為指標的三個維度輸入名稱和字符。如果將維度分配給指標篩選條件建立的指標，則無法為這些指標指派預設值。

> **Note**
>
> 僅在 JSON 或空格分隔指標篩選條件中支援維度。

14. 選擇 Create metric filter (建立指標篩選條件)。可以從導覽窗格中找到您建立的指標篩選條件。選擇 Logs (日誌)，然後選擇 Log groups (日誌群組)。選擇您為其建立指標篩選條件的日誌群組名稱，然後選取 Metric filters (指標篩選條件) 標籤。

範例：計算日誌事件數量

日誌事件監控的最簡單類型就是計數發生的日誌事件數。您可以這樣做以保留所有事件的計數，以建立「活動訊號」樣式監控或僅練習建立指標篩選條件。

在以下 CLI 範例中，系統會將名為 MyAppAccessCount 的指標篩選條件套用到日誌群組 MyApp/access.log，以在 CloudWatch 命名空間 MyNamespace 中建立指標 EventCount。系統會將篩選條件設定為符合任何日誌事件的內容，並以「1」遞增指標。

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選擇日誌群組的名稱。
4. 選擇 `Actions` > Create metric filter (建立指標篩選條件)。
5. 將 Filter Pattern (篩選條件模式) 和 Select Log Data to Test (選取要測試的日誌資料) 保留空白。
6. 選擇 Next (下一步)，然後針對 Filter Name (篩選條件名稱)，輸入 **`EventCount`**。
7. 在 Metric Details (指標詳細資訊) 下的 Metric Namespace (指標命名空間) 中，輸入 **`MyNameSpace`**。
8. 針對 Metric Name (指標名稱)，輸入 **`MyAppEventCount`**。
9. 確認 Metric Value (指標值) 為 1。這會指定針對每個日誌事件的計數以 1 遞增。
10. 針對 Default Value (預設值)，輸入 0，然後選擇 Next (下一步)。指定預設值可確保即使沒有任何日誌事件發生時仍會報告資料，以避免發生 spotty 指標 (資料有時不存在)。
11. 選擇 Create metric filter (建立指標篩選條件)。

使用 建立指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs put-metric-filter \
  --log-group-name MyApp/access.log \
```

```
  --filter-name EventCount \
  --filter-pattern " " \
  --metric-transformations \
  metricName=MyAppEventCount,metricNamespace=MyNamespace,metricValue=1,defaultValue=0
```

您可以透過張貼任何事件資料來測試這個新政策。您應該會看到發佈到指標 MyAppAccessEventCount 的資料點。

使用 發佈事件資料 AWS CLI

在命令提示中，執行下列命令：

```
aws logs put-log-events \
  --log-group-name MyApp/access.log --log-stream-name TestStream1 \
  --log-events \
    timestamp=1394793518000,message="Test event 1" \
    timestamp=1394793518000,message="Test event 2" \
    timestamp=1394793528000,message="This message also contains an Error"
```

範例：計算詞彙的出現次數

日誌事件經常包含您想要計數，或是有關操作成功或失敗操作的重要訊息。例如，若指定的操作失敗，錯誤可能會發生且系統會將該錯誤記錄到日誌檔。您可能想要監控這些項目，以了解錯誤的趨勢。

在下例中，建立指標篩選條件來監控「Error」詞彙。政策已建立並新增到日誌群組 MyApp/message.log。CloudWatch Logs 會將資料點發佈至 MyApp/message.log 命名空間中的 CloudWatch 自訂指標 ErrorCount，針對每個包含 Error 的事件，值為 "1"。如果事件不包含單字「Error」，則會發佈值 0。在 CloudWatch 主控台中製作此資料的圖表時，請務必使用總和統計。

建立指標篩選條件後，您可以在 CloudWatch 主控台中檢視指標。選取要檢視的指標時，請選取符合日誌群組名稱的指標命名空間。如需詳細資訊，請參閱檢視可用的指標。

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選擇日誌群組的名稱。
4. 選擇 Actions (動作) > Create metric filter (建立指標篩選條件)。
5. 針對 Filter Pattern (篩選條件模式)，輸入 **`Error`**。

> **Note**
>
> 在 Filter Pattern (篩選條件模式) 中的所有項目都會區分大小寫。

6. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件，用以測試模式。每個日誌事件都必須在一行內，因為 Log event messages (日誌事件訊息) 方塊中使用換行來分隔日誌事件。
7. 選擇 Next (下一步)，然後在 Assign metric (指派指標) 頁面上，針對 Filter Name (篩選條件名稱) 輸入 **MyAppErrorCount**。
8. 在 Metric Details (指標詳細資訊) 下的 Metric Namespace (指標命名空間) 方塊中，輸入 MyNameSpace。
9. 針對 Metric Name (指標名稱)，輸入 ErrorCount。
10. 確認 Metric Value (指標值) 為 1。這會指定針對每個包含「Error」的日誌事件計數以 1 的方式遞增。
11. 針對 Default Value (預設值)，輸入 0，然後選擇 Next (下一步)。
12. 選擇 Create metric filter (建立指標篩選條件)。

使用 建立指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs put-metric-filter \
  --log-group-name MyApp/message.log \
  --filter-name MyAppErrorCount \
  --filter-pattern 'Error' \
  --metric-transformations \
      metricName=ErrorCount,metricNamespace=MyNamespace,metricValue=1,defaultValue=0
```

您可以透過張貼在訊息中包含「錯誤」單字的事件來測試這個新政策。

使用 發佈事件 AWS CLI

畫面出現命令提示時，執行下列 命令。請注意，模式會區分大小寫。

```
aws logs put-log-events \
  --log-group-name MyApp/access.log --log-stream-name TestStream1 \
  --log-events \
```

```
    timestamp=1394793518000,message="This message contains an Error" \
    timestamp=1394793528000,message="This message also contains an Error"
```

範例：Count HTTP 404 代碼

您可以使用 CloudWatch Logs 監控 Apache 伺服器傳回 HTTP 404 回應的次數，HTTP 404 回應是找不到頁面的回應碼。您可能想要監控此次數以了解您的網站訪客找不到所需資源的頻率。假設您的日誌記錄的建置是包含每個日誌事件 (網站瀏覽) 的以下資訊：

- 請求者 IP 地址
- RFC 1413 身分
- 使用者名稱
- 時間戳記
- 含請求資源和通訊協定的請求方法
- 要請求的 HTTP 回應碼
- 請求中傳入的位元組數

此範例看起來與以下類似：

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 404 2326
```

您可以指定規則，該規則會嘗試比對該結構的事件是否含 HTTP 404 錯誤，如下範例所示：

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選擇 `Actions` > Create metric filter (建立指標篩選條件)。
4. 針對 Filter Pattern (篩選條件模式)，輸入 **`[IP, UserInfo, User, Timestamp, RequestInfo, StatusCode=404, Bytes]`**。
5. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件，用以測試模式。每個日誌事件都必須在一行內，因為 Log event messages (日誌事件訊息) 方塊中使用換行來分隔日誌事件。
6. 選擇 Next (下一步)，然後針對 Filter Name (篩選條件名稱)，輸入 HTTP404Errors。

7. 在 Metric Details (指標詳細資料) 下的 Metric Namespace (指標命名空間) 中，輸入 **MyNameSpace**。
8. 針對 Metric Name (指標名稱)，輸入 **ApacheNotFoundErrorCount**。
9. 確認 Metric Value (指標值) 為 1。這會指定針對每個 404 錯誤事件的計數增加 1。
10. 針對 Default Value (預設值)，輸入 0，然後選擇 Next (下一步)。
11. 選擇 Create metric filter (建立指標篩選條件)。

使用 建立指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs put-metric-filter \
  --log-group-name MyApp/access.log \
  --filter-name HTTP404Errors \
  --filter-pattern '[ip, id, user, timestamp, request, status_code=404, size]' \
  --metric-transformations \
      metricName=ApacheNotFoundErrorCount,metricNamespace=MyNamespace,metricValue=1
```

在這個範例中，會使用到左右方括號、雙引號和字元字串 404 等常值字元。此模式需與被視為監控之日誌事件的整個日誌事件訊息相比對。

您可以使用 describe-metric-filters 命令來驗證指標篩選條件的建立。您應該會看到類似下面的輸出：

```
aws logs describe-metric-filters --log-group-name MyApp/access.log

{
    "metricFilters": [
        {
            "filterName": "HTTP404Errors",
            "metricTransformations": [
                {
                    "metricValue": "1",
                    "metricNamespace": "MyNamespace",
                    "metricName": "ApacheNotFoundErrorCount"
                }
            ],
            "creationTime": 1399277571078,
            "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
 size]"
        }
```

```
    ]
}
```

現在您可以手動張貼幾個事件：

```
aws logs put-log-events \
--log-group-name MyApp/access.log --log-stream-name hostname \
--log-events \
timestamp=1394793518000,message="127.0.0.1 - bob [10/Oct/2000:13:55:36 -0700] \"GET /
apache_pb.gif HTTP/1.0\" 404 2326" \
timestamp=1394793528000,message="127.0.0.1 - bob [10/Oct/2000:13:55:36 -0700] \"GET /
apache_pb2.gif HTTP/1.0\" 200 2326"
```

放置這些範例日誌事件後，您可以擷取在 CloudWatch 主控台中名為 ApacheNotFoundErrorCount 的指標。

範例：Count HTTP 4xx 代碼

在上述範例中，您可能想要監控 web 服務存取日誌和監控 HTTP 回應碼層級。例如，您可能想要監控所有 HTTP 400 層級錯誤。不過，您可能不會想要為每個傳回程式碼指定新指標篩選條件。

以下範例示範如何建立指標，其中包含使用從 範例：Count HTTP 404 代碼 範例之 Apache 存取日誌格式來自存取日誌的所有 400 層級 HTTP 程式碼回應。

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選擇 Apache 伺服器的日誌群組名稱。
4. 選擇 `Actions` > Create metric filter (建立指標篩選條件)。
5. 針對 Filter Pattern (篩選條件模式)，輸入 **`[ip, id, user, timestamp, request, status_code=4*, size]`**。
6. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件，用以測試模式。每個日誌事件都必須在一行內，因為 Log event messages (日誌事件訊息) 方塊中使用換行來分隔日誌事件。
7. 選擇 Next (下一步)，然後針對 Filter Name (篩選條件名稱)，輸入 **`HTTP4xxErrors`**。
8. 在 Metric (指標詳細資訊) 下的 Metric Namespace (指標命名空間) 中，輸入 **`MyNameSpace`**。
9. 針對 Metric name (指標名稱)，輸入 HTTP4xxErrors。

10. 針對 Metric value (指標值)，輸入 1。這會指定針對每個包含「4xx 錯誤」的日誌事件以 1 的方式遞增計數。
11. 針對 Default value (預設值)，輸入 0，然後選擇 Next (下一步)。
12. 選擇 Create metric filter (建立指標篩選條件)。

使用 建立指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs put-metric-filter \
  --log-group-name MyApp/access.log \
  --filter-name HTTP4xxErrors \
  --filter-pattern '[ip, id, user, timestamp, request, status_code=4*, size]' \
  --metric-transformations \
  metricName=HTTP4xxErrors,metricNamespace=MyNamespace,metricValue=1,defaultValue=0
```

您可以使用 put-event 呼叫中的以下資料來測試這個規則。如果您沒有在之前的範例中移除監控規則，您將會產生兩種不同的指標。

```
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /~test/ HTTP/1.1" 200 3
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:51:34 -0700] "GET /~test/index.html HTTP/1.1" 200 3
```

範例：從 Apache 日誌擷取欄位並指派維度

有時，不使用計數，而是在指標值的個別日誌事件中使用值很有用。此範例顯示如何建立擷取規則來建立指標，該指標會量測 Apache Web 伺服器傳出的位元組數。

此範例也會說明如何將維度指派給您要建立的指標。

使用 CloudWatch 主控台建立指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選擇 Apache 伺服器的日誌群組名稱。
4. 選擇 `Actions` > Create metric filter (建立指標篩選條件)。

5. 針對 Filter Pattern (篩選條件模式)，輸入 **[ip, id, user, timestamp, request, status_code, size]**。
6. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件，用以測試模式。每個日誌事件都必須在一行內，因為 Log event messages (日誌事件訊息) 方塊中使用換行來分隔日誌事件。
7. 選擇 Next (下一步)，然後針對 Filter Name (篩選條件名稱)，輸入 **size**。
8. 在 Metric (指標詳細資訊) 下的 Metric Namespace (指標命名空間) 中，輸入 **MyNameSpace**。因為這是新的命名空間，請務必選取 Create new (新建)。
9. 針對 Metric name (指標名稱)，輸入 **BytesTransferred**
10. 針對 Metric value (指標值)，輸入 **$size**。
11. 針對 Unit (單位)，選取Bytes (位元組)。
12. 針對 Dimension Name (維度名稱)，輸入 **IP**。
13. 針對 Dimension Value (維度值)，輸入 **$ip**，然後選擇 Next (下一步)。
14. 選擇 Create metric filter (建立指標篩選條件)。

使用 建立此指標篩選條件 AWS CLI

在命令提示中，執行下列命令

```
aws logs put-metric-filter \
--log-group-name MyApp/access.log \
 --filter-name BytesTransferred \
 --filter-pattern '[ip, id, user, timestamp, request, status_code, size]' \
 --metric-transformations  \
 metricName=BytesTransferred,metricNamespace=MyNamespace,metricValue='$size'
```

```
aws logs put-metric-filter \
--log-group-name MyApp/access.log \
--filter-name BytesTransferred \
--filter-pattern '[ip, id, user, timestamp, request, status_code, size]' \
--metric-transformations  \
metricName=BytesTransferred,metricNamespace=MyNamespace,metricValue='$size',unit=Bytes,dimensio
$ip}}'
```

> **Note**
>
> 在此命令中，使用此格式指定多個維度。

```
aws logs put-metric-filter \
--log-group-name my-log-group-name \
--filter-name my-filter-name \
--filter-pattern 'my-filter-pattern' \
--metric-transformations  \
metricName=my-metric-name,metricNamespace=my-metric-namespace,metricValue=my-token,unit=unit,dimensions='{dimension1=$dim,dimension2=$dim2,dim3=$dim3}'
```

您可以使用 put-log-event 呼叫中的以下資料來測試這個規則。如果您沒有在之前的範例中移除監控規則，此將會產生兩種不同的指標。

```
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /~test/ HTTP/1.1" 200 3
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:51:34 -0700] "GET /~test/index.html HTTP/1.1" 200 3
```

列出指標篩選條件

您可以列出日誌群組中所有指標篩選條件。

使用 CloudWatch 主控台列出指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 在日誌群組清單的內容窗格中，選擇在 Metric Filters (指標篩選條件) 欄中的篩選條件數。

   Log Groups > Filters for (日誌群組 > 篩選條件) 畫面會列出與日誌群組相關聯的所有指標篩選條件。

使用 列出指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs describe-metric-filters --log-group-name MyApp/access.log
```

下列為範例輸出：

```
{
    "metricFilters": [
        {
            "filterName": "HTTP404Errors",
            "metricTransformations": [
                {
                    "metricValue": "1",
                    "metricNamespace": "MyNamespace",
                    "metricName": "ApacheNotFoundErrorCount"
                }
            ],
            "creationTime": 1399277571078,
            "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
 size]"
        }
    ]
}
```

刪除指標篩選條件

您可透過政策名稱和其所屬的日誌群組來辨識該政策。

使用 CloudWatch 主控台刪除指標篩選條件

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 在內容窗格的 Metric Filter (指標篩選條件) 欄中，選擇日誌群組的指標篩選條件數。
4. 在 Metric Filters (指標篩選條件) 畫面下，針對您要刪除的篩選條件，選取其名稱右側的核取方塊。然後選擇刪除。
5. 出現確認提示時，請選擇刪除。

使用 刪除指標篩選條件 AWS CLI

在命令提示中，執行下列命令：

```
aws logs delete-metric-filter --log-group-name MyApp/access.log \
 --filter-name MyFilterName
```

使用訂閱即時處理日誌資料

您可以使用訂閱從 CloudWatch Logs 存取日誌事件的即時摘要，並將其交付至其他服務，例如 Amazon Kinesis 串流、Amazon Data Firehose 串流，或 AWS Lambda 用於自訂處理、分析或載入至其他系統。日誌事件傳送至接收端服務時會以 Base64 編碼並以 gzip 格式壓縮。

您也可以使用 CloudWatch Logs 集中，將多個帳戶和區域的日誌資料複寫至中央位置。如需詳細資訊，請參閱跨帳戶跨區域日誌集中化。

若要開始訂閱日誌事件，請建立接收資源，例如將交付事件的 Amazon Kinesis Data Streams 串流。訂閱篩選條件定義篩選條件模式，用於篩選哪些日誌事件交付至您的 AWS 資源，以及有關將相符日誌事件傳送到何處的資訊。日誌事件會在擷取後立即傳送到接收資源，通常不到三分鐘。

> **Note**
>
> 如果具有訂閱的日誌群組使用日誌轉換，則會將篩選條件模式與日誌事件的轉換版本進行比較。如需詳細資訊，請參閱在擷取期間轉換日誌。

您可以在帳戶層級和日誌群組層級建立訂閱。每個帳戶在每個區域可以有一個帳戶層級訂閱篩選條件。每個日誌群組最多有兩個相關聯的訂閱篩選條件。

> **Note**
>
> 如果目的地服務傳回可重試的錯誤，例如調節例外狀況或可重試的服務例外狀況 (例如 HTTP 5xx)，CloudWatch Logs 會繼續重試傳送，最多 24 小時。如果錯誤是不可重試的錯誤，例如 AccessDeniedException 或 ResourceNotFoundException，CloudWatch Logs 不會嘗試重新交付。在這些情況下，訂閱篩選條件會停用最多 10 分鐘，然後 CloudWatch Logs 會重試將日誌傳送至目的地。在此停用期間，日誌會略過。

CloudWatch Logs 也會產生有關將日誌事件轉送到訂閱的 CloudWatch 指標。如需詳細資訊，請參閱使用 CloudWatch 指標監控使用量。

您也可以使用 CloudWatch Logs 訂閱，將日誌資料以近乎即時的方式串流到 Amazon OpenSearch Service 叢集。如需詳細資訊，請參閱將 CloudWatch Logs 資料串流到 Amazon OpenSearch Service。

只有標準日誌類別中的日誌群組才支援訂閱。如需日誌類別的詳細資訊，請參閱 日誌類別。

> **Note**
>
> 訂閱篩選條件可能會批次記錄事件，以最佳化傳輸並減少對目的地進行的呼叫量。不保證批次處理，但盡可能使用。

若要依排程批次處理和分析日誌資料，請考慮使用 使用排程查詢自動化日誌分析。排程查詢會自動執行 CloudWatch Logs Insights 查詢，並將結果交付至目的地，例如 Amazon S3 儲存貯體或 Amazon EventBridge 事件匯流排。

> **Note**
>
> 訂閱篩選條件可確保事件至少交付一次，而重複的事件偶爾可能會發生。

目錄

概念

每個訂閱篩選條件是由下列關鍵元素組成：

篩選條件模式

CloudWatch Logs 應如何解譯每個日誌事件中資料的符號描述，以及限制傳遞至目的地 AWS 資源內容的篩選表達式。如需篩選條件模式語法的詳細資訊，請參閱 用於指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法。

目的地 ARN

您要用作訂閱摘要目的地之 Amazon Kinesis Data Streams 串流、Firehose 串流或 Lambda 函數的 Amazon Resource Name (ARN)。

角色 ARN

IAM 角色，授予 CloudWatch Logs 必要許可將資料放入所選目的地。Lambda 目的地不需此角色，因為 CloudWatch Logs 可以從 Lambda 函數本身的存取控制設定中取得必要許可。

分佈

當目的地是 Amazon Kinesis Data Streams 中的串流時,用來將日誌資料分送到目的地的方法。在預設情況下，日誌資料是依日誌串流來分組的。如需進行更多分發，您可以將日誌資料隨機分組。

對於日誌群組層級訂閱，也包含下列金鑰元素：

日誌群組名稱

要與訂閱篩選條件關聯的日誌群組。所有上傳到此日誌群組的日誌事件取決於訂閱篩選條件，符合篩選條件的日誌事件會傳送至負責接收相符日誌事件的目的地服務。

對於帳戶層級訂閱，也包含下列金鑰元素：

選擇條件

用於選取哪些日誌群組套用帳戶層級訂閱篩選條件的條件。如果您未指定此項目，帳戶層級訂閱篩選條件會套用至帳戶中的所有日誌群組。此欄位用於防止無限日誌迴圈。如需無限日誌迴圈問題的詳細資訊，請參閱 日誌遞迴預防。

選擇條件的大小限制為 25 KB。

對於集中式日誌群組，也會包含下列金鑰元素。這些元素可以用作欄位選擇條件，以協助識別日誌資料來源，以便更精細地篩選和分析衍生自集中式日誌的指標。

@aws.account

此欄位識別日誌事件來源 AWS 的帳戶 ID。

@aws.region

此欄位識別產生日誌事件 AWS 的區域。

日誌群組層級訂閱篩選條件

您可以搭配 Amazon Kinesis Data Streams AWS Lambda、Amazon Data Firehose 或 Amazon OpenSearch Service 使用訂閱篩選條件。透過訂閱篩選條件傳送到服務的日誌會以 gzip 格式進行 base64 編碼和壓縮。如果您搭配 使用集中式日誌 AWS Organizations，您可以選擇發出 `@aws.account`和 `@aws.region` 系統欄位，以識別哪些資料來自組織中的帳戶和區域。本節提供您可以遵循的範例，以建立 CloudWatch Logs 訂閱篩選條件，將日誌資料傳送至 Firehose、Lambda、Amazon Kinesis Data Streams 和 OpenSearch Service。

> **Note**
>
> 如果您想要搜尋日誌資料，請參閱篩選和模式語法。

範例

- 範例 1：使用 Amazon Kinesis Data Streams 的訂閱篩選條件
- 範例 2：使用 的訂閱篩選條件 AWS Lambda
- 範例 3：使用 Amazon Data Firehose 的訂閱篩選條件
- 範例 4：使用 Amazon OpenSearch Service 的訂閱篩選條件

範例 1：使用 Amazon Kinesis Data Streams 的訂閱篩選條件

下列範例將訂閱篩選條件與包含 AWS CloudTrail 事件的日誌群組建立關聯。訂閱篩選條件會將 "Root" AWS credentials 所做的每個記錄活動交付至 Amazon Kinesis Data Streams 中名為 "RootAccess." 如需如何將 AWS CloudTrail 事件傳送至 CloudWatch Logs 的詳細資訊，請參閱AWS CloudTrail 《使用者指南》中的將 CloudTrail 事件傳送至 CloudWatch Logs。

> **Note**
>
> 在您建立 串流前，計算將產生的日誌資料磁碟區。請務必使用足夠碎片建立 串流，以處理此磁碟區。如果串流沒有足夠的碎片，日誌串流將受到限制。如需更多有關串流磁碟區限制的資訊，請參閱配額與限制。
> 限流的交付項目會持續重試，時間長達 24 小時。24 小時後，失敗的交付項目就會捨棄。
> 若要降低限流風險，您可以採取下步驟：

- 當您使用 PutSubscriptionFilter 或 put-subscription-filter 建立訂閱篩選條件`distribution`時，請指定 `random`的 。根據預設，串流篩選條件分佈是依日誌串流，這可能會導致限流。
- 使用 CloudWatch 指標監控您的串流。如此可協助您找出任何限流，並根據實際情況調整您的組態。例如，將資料轉送至訂閱目的地時，`DeliveryThrottling` 指標可用來追蹤 CloudWatch Logs 遭限流的日誌事件數量。如需監控的詳細資訊，請參閱使用 CloudWatch 指標監控使用量。
- 在 Amazon Kinesis Data Streams 中為您的串流使用隨需容量模式。隨需模式會在您的工作負載上升或下降時，立即為您的工作負載調整所需的容量。有關隨需容量模式的詳細資訊，請參閱隨需模式。
- 限制您的 CloudWatch 訂閱篩選條件模式，以符合 Amazon Kinesis Data Streams 中的串流容量。如果您傳送太多資料至串流，您可能需要減少篩選條件大小或調整篩選條件標準。

建立 Amazon Kinesis Data Streams 的訂閱篩選條件

1. 使用下列命令建立目的地 串流：

```
$ C:\>  aws kinesis create-stream --stream-name "RootAccess" --shard-count 1
```

2. 等到 串流成為作用中 (這可能需要花費幾分鐘)。您可以使用下列 Amazon Kinesis Data Streams describe-stream 命令來檢查 StreamDescription.StreamStatus 屬性。此外，請留意 StreamDescription.StreamARN 值，因您將在後續步驟需要此值：

```
aws kinesis describe-stream --stream-name "RootAccess"
```

下列為範例輸出：

```
{
    "StreamDescription": {
        "StreamStatus": "ACTIVE",
        "StreamName": "RootAccess",
        "StreamARN": "arn:aws:kinesis:us-east-1:123456789012:stream/RootAccess",
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "EndingHashKey": "340282366920938463463374607431768211455",
```

```
                    "StartingHashKey": "0"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
                    "495511352186888184566795038319814587845913527021815726l0"
                }
            }
        ]
    }
}
```

3. 建立將授予 CloudWatch Logs 許可以將資料放到串流中的 IAM 角色。首先，您將需要在檔案中建立信任政策 (例如，~/TrustPolicyForCWL-Kinesis.json)。請使用文字編輯器來建立此政策。請勿使用 IAM 主控台建立這一項。

   此政策包含 aws:SourceArn 全域條件內容金鑰，以協助預防混淆代理人安全問題。如需詳細資訊，請參閱預防混淆代理人。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請注意傳回的 Role.Arn 值，因您將在後續步驟需要此值：

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

   以下為輸出範例。

```
{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
```

```
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": { "arn:aws:logs:region:123456789012:*" }
                    }
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
    }
}
```

5. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，您將需要在檔案中建立許可政策 (例如，~/PermissionsForCWL-Kinesis.json)。請使用文字編輯器來建立此政策。請勿使用 IAM 主控台建立這一項。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/RootAccess"
    }
  ]
}
```

6. 使用以下 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. 在串流處於作用中狀態且您已建立 IAM 角色後，您就可以建立 CloudWatch Logs 訂閱篩選條件。訂閱篩選條件會立即開始將即時日誌資料從所選的日誌群組傳送到 串流：

```
aws logs put-subscription-filter \
    --log-group-name "CloudTrail/logs" \
```

```
    --filter-name "RootAccess" \
    --filter-pattern "{$.userIdentity.type = Root}" \
    --destination-arn "arn:aws:kinesis:region:123456789012:stream/RootAccess" \
    --role-arn "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
```

8. 在您設定訂閱篩選條件後，CloudWatch Logs 會將所有符合篩選條件模式的傳入日誌事件轉送至串流。您可以透過抓取 Amazon Kinesis Data Streams 碎片迭代器，並使用 Amazon Kinesis Data Streams get-records 命令擷取一些 Amazon Kinesis Data Streams 記錄，來驗證是否發生這種情況：

```
aws kinesis get-shard-iterator --stream-name RootAccess --shard-id
 shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
```

請注意，您可能需要在 Amazon Kinesis Data Streams 開始傳回資料之前進行此呼叫幾次。

您應該預期會看到含一系列的記錄的回應。Amazon Kinesis Data Streams 記錄中的資料屬性是 base64 編碼並以 gzip 格式壓縮。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "owner": "111111111111",
    "logGroup": "CloudTrail/logs",
```

```
    "logStream": "111111111111_CloudTrail/logs_us-east-1",
    "subscriptionFilters": [
        "Destination"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424298221568",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221569",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221570",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        }
    ]
}
```

在上述資料結構的關鍵元素如下：

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

messageType

資料訊息將使用「DATA_MESSAGE」類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Amazon Kinesis Data Streams 記錄，主要用於檢查目的地是否可連線。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。「id」屬性是每個記錄事件的唯一識別符。

範例 2：使用 的訂閱篩選條件 AWS Lambda

在此範例中，您將建立 CloudWatch Logs 訂閱篩選條件，將日誌資料傳送至您的 AWS Lambda 函數。

> **Note**
>
> 建立 Lambda 函數前，請計算將產生的日誌資料量。請務必建立可以處理此磁碟區的函數。如果函數沒有足夠的磁碟區，日誌串流將受到限制。如需 Lambda 限制的詳細資訊，請參閱 AWS Lambda 限制。

建立 Lambda 的訂閱篩選條件

1. 建立 AWS Lambda 函數。

   確保您已設定 Lambda 執行角色。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的步驟 2.2：建立 IAM 角色 (執行角色)。

2. 開啟文字編輯器，並建立名為 `helloWorld.js` 的檔案，內含下列內容：

```
var zlib = require('zlib');
exports.handler = function(input, context) {
    var payload = Buffer.from(input.awslogs.data, 'base64');
    zlib.gunzip(payload, function(e, result) {
        if (e) {
            context.fail(e);
        } else {
            result = JSON.parse(result.toString());
            console.log("Event Data:", JSON.stringify(result, null, 2));
            context.succeed();
```

```
        }
    });
};
```

3. 壓縮檔案 helloWorld.js，並以名稱 helloWorld.zip 將其儲存。
4. 使用下列命令，其中角色是您在第一個步驟中設定的 Lambda 執行角色：

```
aws lambda create-function \
    --function-name helloworld \
    --zip-file fileb://file-path/helloWorld.zip \
    --role lambda-execution-role-arn \
    --handler helloWorld.handler \
    --runtime nodejs12.x
```

5. 授予 CloudWatch Logs 許可來執行函數。使用下列命令，將預留位置帳戶取代為您自己的帳戶且將預留位置日誌群組取代為要處理的日誌群組：

```
aws lambda add-permission \
    --function-name "helloworld" \
    --statement-id "helloworld" \
    --principal "logs.amazonaws.com" \
    --action "lambda:InvokeFunction" \
    --source-arn "arn:aws:logs:region:123456789123:log-group:TestLambda:*" \
    --source-account "123456789012"
```

6. 使用下列命令建立訂閱篩選條件，將預留位置帳戶取代為您自己的帳戶且將預留位置日誌群組取代為要處理的日誌群組：

```
aws logs put-subscription-filter \
    --log-group-name myLogGroup \
    --filter-name demo \
    --filter-pattern "" \
    --destination-arn arn:aws:lambda:region:123456789123:function:helloworld
```

7. (選用) 使用範例日誌事件進行測試。在命令提示字元中執行下列命令，這會將簡單日誌訊息放置到訂閱的串流。

   若要查看 Lambda 函數的輸出，請導覽至 Lambda 函數，其中您將會在 /aws/lambda/helloworld 中看到輸出：

```
aws logs put-log-events --log-group-name myLogGroup --log-stream-name stream1 --
log-events "[{\"timestamp\":<CURRENT TIMESTAMP MILLIS> , \"message\": \"Simple
 Lambda Test\"}]"
```

預期會看到含一系列 Lambda 的回應。Lambda 記錄中的 Data (資料) 屬性是以 Base64 編碼並以 gzip 格式壓縮。Lambda 收到的實際酬載為以下格式：`{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }`。您可以從命令列使用以下 Unix 命令來檢視原始資料：

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "owner": "123456789012",
    "logGroup": "CloudTrail",
    "logStream": "123456789012_CloudTrail_us-east-1",
    "subscriptionFilters": [
        "Destination"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424298221568",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221569",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221570",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        }
```

```
    ]
}
```

在上述資料結構的關鍵元素如下：

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

messageType

資料訊息將使用「DATA_MESSAGE」類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Lambda 記錄，主要用於檢查是否可到達目的地。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。「id」屬性是每個記錄事件的唯一識別符。

範例 3：使用 Amazon Data Firehose 的訂閱篩選條件

在此範例中，您將建立 CloudWatch Logs 訂閱，將任何符合您定義篩選條件的傳入日誌事件傳送至 Amazon Data Firehose 交付串流。從 CloudWatch Logs 傳送至 Amazon Data Firehose 的資料已使用 gzip 第 6 級壓縮進行壓縮，因此您不需要在 Firehose 交付串流中使用壓縮。然後，您可以使用 Firehose 中的解壓縮功能自動解壓縮日誌。如需詳細資訊，請參閱將 CloudWatch Logs 傳送至 Firehose。

> **Note**
>
> 建立 Firehose 串流之前，請先計算要產生的日誌資料量。請務必建立可處理此磁碟區的 Firehose 串流。如果串流無法處理磁碟區、日誌串流將受到限制。如需 Firehose 串流磁碟區限制的詳細資訊，請參閱 Amazon Data Firehose Data Limits。

建立 Firehose 的訂閱篩選條件

1. 建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體。我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，請跳到步驟 2。

   執行以下命令，將預留位置 Region 換成您想要使用的區域：

   ```
   aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration
    LocationConstraint=region
   ```

   下列為範例輸出：

   ```
   {
       "Location": "/amzn-s3-demo-bucket2"
   }
   ```

2. 建立 IAM 角色，授予 Amazon Data Firehose 將資料放入 Amazon S3 儲存貯體的許可。

   如需詳細資訊，請參閱《Amazon Data Firehose 開發人員指南》中的使用 Amazon Data Firehose 控制存取。

   首先，請按如下所示，使用文字編輯器來建立檔案 ~/TrustPolicyForFirehose.json 中的信任政策：

   ```
   {
     "Statement": {
       "Effect": "Allow",
       "Principal": { "Service": "firehose.amazonaws.com" },
       "Action": "sts:AssumeRole"
       }
   }
   ```

3. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請注意傳回的 Role.Arn 值，因您將在後續步驟需要此值：

   ```
   aws iam create-role \
    --role-name FirehosetoS3Role \
    --assume-role-policy-document file://~/TrustPolicyForFirehose.json

   {
       "Role": {
   ```

```
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "firehose.amazonaws.com"
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "FirehosetoS3Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/FirehosetoS3Role"
    }
}
```

4. 建立許可政策，以定義 Firehose 可以在您的帳戶上執行的動作。首先，使用文字編輯器來建立檔案 ~/PermissionsForFirehose.json 中的許可政策：

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "s3:AbortMultipartUpload",
          "s3:GetBucketLocation",
          "s3:GetObject",
          "s3:ListBucket",
          "s3:ListBucketMultipartUploads",
          "s3:PutObject" ],
      "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket2",
          "arn:aws:s3:::amzn-s3-demo-bucket2/*" ]
    }
  ]
}
```

5. 使用以下 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. 如下所示建立目的地 Firehose 交付串流，將 RoleARN 和 BucketARN 的預留位置值取代為您建立的角色和儲存貯體 ARNs：

```
aws firehose create-delivery-stream \
   --delivery-stream-name 'my-delivery-stream' \
   --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::123456789012:role/FirehosetoS3Role", "BucketARN":
 "arn:aws:s3:::amzn-s3-demo-bucket2"}'
```

請注意，Firehose 會自動為交付的 Amazon S3 物件使用 YYYY/MM/DD/HH UTC 時間格式的字首。您可以指定在時間格式前綴前要新增的額外前綴。如果字首結尾是斜線 (/)，則會在 Amazon S3 儲存貯體中顯示為資料夾。

7. 等到串流成為作用中 (這可能需要花費幾分鐘)。您可以使用 Firehose describe-delivery-stream 命令來檢查 DeliveryStreamDescription.DeliveryStreamStatus 屬性。此外，請留意 DeliveryStreamDescription.DeliveryStreamARN 值，因您將在後續步驟需要此值：

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
{
    "DeliveryStreamDescription": {
        "HasMoreDestinations": false,
        "VersionId": "1",
        "CreateTimestamp": 1446075815.822,
        "DeliveryStreamARN": "arn:aws:firehose:us-
east-1:123456789012:deliverystream/my-delivery-stream",
        "DeliveryStreamStatus": "ACTIVE",
        "DeliveryStreamName": "my-delivery-stream",
        "Destinations": [
            {
                "DestinationId": "destinationId-000000000001",
                "S3DestinationDescription": {
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "RoleARN": "delivery-stream-role",
                    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket2",
                    "BufferingHints": {
                        "IntervalInSeconds": 300,
                        "SizeInMBs": 5
                    }
```

```
                }
            }
        ]
    }
}
```

8. 建立 IAM 角色，授予 CloudWatch Logs 將資料放入 Firehose 交付串流的許可。首先，使用文字編輯器來建立檔案 ~/TrustPolicyForCWL.json 中的信任政策：

   此政策包含 aws:SourceArn 全域條件內容金鑰，以協助預防混淆代理人安全問題。如需詳細資訊，請參閱預防混淆代理人。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
         "StringLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
     }
  }
}
```

9. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請注意傳回的 Role.Arn 值，因您將在後續步驟需要此值：

```
aws iam create-role \
--role-name CWLtoKinesisFirehoseRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Condition": {
                     "StringLike": {
```

```
                        "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
                    }
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
    }
}
```

10. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器來建立許可政策檔案 (例如，~/PermissionsForCWL.json)：

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-name"]
      }
    ]
}
```

11. 使用 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. 在 Amazon Data Firehose 交付串流處於作用中狀態且您已建立 IAM 角色後，您可以建立 CloudWatch Logs 訂閱篩選條件。訂閱篩選條件會立即開始從所選日誌群組到 Amazon Data Firehose 交付串流的即時日誌資料流程：

```
aws logs put-subscription-filter \
    --log-group-name "CloudTrail" \
    --filter-name "Destination" \
    --filter-pattern "{$.userIdentity.type = Root}" \
    --destination-arn "arn:aws:firehose:region:123456789012:deliverystream/my-delivery-stream" \
```

```
    --role-arn "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
```

13. 設定訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式的所有傳入日誌事件轉送至您的 Amazon Data Firehose 交付串流。您的資料會根據 Amazon Data Firehose 交付串流上設定的時間緩衝間隔，開始出現在 Amazon S3 中。一旦經過足夠的時間，您就可以檢查 Amazon S3 儲存貯體來驗證資料。

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket2' --prefix 'firehose/'
{
    "Contents": [
        {
            "LastModified": "2015-10-29T00:01:25.000Z",
            "ETag": "\"a14589f8897f4089d3264d9e2d1f1610\"",
            "StorageClass": "STANDARD",
            "Key": "firehose/2015/10/29/00/my-delivery-stream-2015-10-29-00-01-21-
a188030a-62d2-49e6-b7c2-b11f1a7ba250",
            "Owner": {
                "DisplayName": "cloudwatch-logs",
                "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b5"
            },
            "Size": 593
        },
        {
            "LastModified": "2015-10-29T00:35:41.000Z",
            "ETag": "\"a7035b65872bb2161388ffb63dd1aec5\"",
            "StorageClass": "STANDARD",
            "Key": "firehose/2015/10/29/00/my-delivery-
stream-2015-10-29-00-35-40-7cc92023-7e66-49bc-9fd4-fc9819cc8ed3",
            "Owner": {
                "DisplayName": "cloudwatch-logs",
                "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b6"
            },
            "Size": 5752
        }
    ]
}
```

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket2' --key 'firehose/2015/10/29/00/
my-delivery-stream-2015-10-29-00-01-21-a188030a-62d2-49e6-b7c2-b11f1a7ba250'
 testfile.gz

{
```

```
    "AcceptRanges": "bytes",
    "ContentType": "application/octet-stream",
    "LastModified": "Thu, 29 Oct 2015 00:07:06 GMT",
    "ContentLength": 593,
    "Metadata": {}
}
```

在 Amazon S3 物件中的資料會以 gzip 格式壓縮。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
zcat testfile.gz
```

範例 4：使用 Amazon OpenSearch Service 的訂閱篩選條件

在此範例中，您將建立 CloudWatch Logs 訂閱，將符合您所定義篩選條件的傳入日誌事件傳送至 OpenSearch Service 網域。

建立 OpenSearch Service 的訂閱篩選條件

1. 建立 OpenSearch Service 網域 如需詳細資訊，請參閱建立 OpenSearch Service 網域
2. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
3. 在導覽窗格中，選擇 Log groups (日誌群組)。
4. 選取日誌群組的名稱。
5. 選擇 Actions (動作)、Subscription filters (訂閱篩選條件)、Create Amazon OpenSearch Service subscription filter (建立 Amazon OpenSearch Service 訂閱篩選條件)。
6. 選擇您是否要串流至此帳戶或其他帳戶中的叢集。
   - 如果您選擇此帳戶，請選取您在步驟 1 中建立的網域。
   - 如果您選擇另一個帳戶，請輸入該網域的 ARN 和端點。
7. 如果您選擇其他帳戶，請提供網域 ARN 和端點。
8. 針對 Amazon OpenSearch Service 叢集，選擇將交付日誌群組資料的叢集名稱
9. 選擇日誌格式。
10. 針對訂閱篩選條件模式，輸入要在日誌事件中找到的術語或模式。這可確保您只將感興趣的資料傳送至 OpenSearch Service 叢集。如需詳細資訊，請參閱指標篩選條件的篩選條件模式語法。
11. (選用) 針對 Select log data to test (選取要測試的日誌資料)，選擇一個日誌串流，然後選擇 Test pattern (測試模式)，以驗證您的搜尋篩選條件是否會傳回您預期的結果。

12. 選擇 Start streaming (開始串流)。

帳戶層級訂閱篩選條件

> **Important**
>
> 存在使用訂閱篩選條件造成無限遞迴迴圈的風險，如果未解決，可能會導致擷取計費大幅增加。為了降低此風險，建議您在帳戶層級訂閱篩選條件中使用選擇條件，以排除從訂閱交付工作流程一部分的資源擷取日誌資料的日誌群組。如需此問題和決定要排除哪些日誌群組的詳細資訊，請參閱 日誌遞迴預防。

您可以設定帳戶層級訂閱政策，其中包含帳戶中的日誌群組子集。帳戶訂閱政策可以使用 Amazon Kinesis Data Streams AWS Lambda或 Amazon Data Firehose。透過帳戶層級訂閱政策傳送至服務的日誌會以 gzip 格式進行 base64 編碼和壓縮。本節提供您可以遵循的範例，為 Amazon Kinesis Data Streams、Lambda 和 Firehose 建立帳戶層級訂閱。

> **Note**
>
> 若要檢視您帳戶中所有訂閱篩選條件政策的清單，請使用 `--policy-type` 參數值為 `SUBSCRIPTION_FILTER_POLICY`的 `describe-account-policies`命令。如需詳細資訊，請參閱 describe-account-policies¶。

範例

- 範例 1：使用 Amazon Kinesis Data Streams 的訂閱篩選條件
- 範例 2：使用 的訂閱篩選條件 AWS Lambda
- 範例 3：使用 Amazon Data Firehose 的訂閱篩選條件

範例 1：使用 Amazon Kinesis Data Streams 的訂閱篩選條件

在您建立要與帳戶層級訂閱政策搭配使用的 Amazon Kinesis Data Streams 資料串流之前，請計算將產生的日誌資料量。請務必使用足夠碎片建立 串流，以處理此磁碟區。如果串流沒有足夠的碎片，則會調節。如需串流磁碟區限制的詳細資訊，請參閱 Amazon Kinesis Data Streams 文件中的配額和限制。

> **Warning**
>
> 由於多個日誌群組的日誌事件會轉送至目的地，因此存在限流風險。限流的交付項目會持續重試，時間長達 24 小時。24 小時後，失敗的交付項目就會捨棄。
> 若要降低限流風險，您可以採取下步驟：
>
> - 使用 CloudWatch 指標監控 Amazon Kinesis Data Streams 串流。這可協助您識別限流，並相應地調整您的組態。例如， `DeliveryThrottling` 指標會追蹤 CloudWatch Logs 在轉送資料至訂閱目的地時受到調節的日誌事件數量。如需詳細資訊，請參閱使用 CloudWatch 指標監控使用量。
> - 在 Amazon Kinesis Data Streams 中為您的串流使用隨需容量模式。隨需模式會在您的工作負載上升或下降時，立即為您的工作負載調整所需的容量。如需詳細資訊，請參閱隨需模式。
> - 限制 CloudWatch Logs 訂閱篩選條件模式，以符合 Amazon Kinesis Data Streams 中的串流容量。如果您傳送太多資料至串流，您可能需要減少篩選條件大小或調整篩選條件標準。

下列範例使用帳戶層級訂閱政策，將所有日誌事件轉送至 Amazon Kinesis Data Streams 中的串流。篩選條件模式會將任何日誌事件與文字相符，`Test`並將其轉送至 Amazon Kinesis Data Streams 中的串流。

為 Amazon Kinesis Data Streams 建立帳戶層級訂閱政策

1. 使用下列命令建立目的地 串流：

```
$ C:\>  aws kinesis create-stream –stream-name "TestStream" –shard-count 1
```

2. 等待幾分鐘讓串流變成作用中。您可以使用 describe-stream 命令來檢查 StreamDescription.StreamStatus 屬性，以驗證串流是否處於作用中狀態。

```
aws kinesis describe-stream --stream-name "TestStream"
```

下列為範例輸出：

```
{
    "StreamDescription": {
        "StreamStatus": "ACTIVE",
        "StreamName": "TestStream",
        "StreamARN": "arn:aws:kinesis:region:123456789012:stream/TestStream",
```

```
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "EndingHashKey": "EXAMPLE84634633746074317682114555",
                    "StartingHashKey": "0"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
                    "EXAMPLE6888184566795038319814587845913527021815726l0"
                }
            }
        ]
    }
}
```

3. 建立將授予 CloudWatch Logs 許可以將資料放到串流中的 IAM 角色。首先，您將需要在檔案中建立信任政策 (例如，~/TrustPolicyForCWL-Kinesis.json)。請使用文字編輯器來建立此政策。

   此政策包含 aws:SourceArn 全域條件內容金鑰，以協助預防混淆代理人安全問題。如需詳細資訊，請參閱預防混淆代理人。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請注意傳回的 Role.Arn 值，因您將在後續步驟需要此值：

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

   以下為輸出範例。

```
{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": { "arn:aws:logs:region:123456789012:*" }
                    }
                }
            }
        },
        "RoleId": "EXAMPLE450GAB4HC5F431",
        "CreateDate": "2023-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
    }
}
```

5. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，您將需要在檔案中建立許可政策 (例如，~/PermissionsForCWL-Kinesis.json)。請使用文字編輯器來建立此政策。請勿使用 IAM 主控台來建立它。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/TestStream"
    }
  ]
}
```

6. 使用以下 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. 在串流處於作用中狀態且您已建立 IAM 角色之後，您可以建立 CloudWatch Logs 訂閱篩選條件政策。此政策會立即開始將即時日誌資料傳送至串流的流程。在此範例中，所有包含字串的日誌事件ERROR都會串流，但名為 LogGroupToExclude1和 的日誌群組中的日誌事件除外LogGroupToExclude2。

```
aws logs put-account-policy \
    --policy-name "ExamplePolicy" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document '{"RoleArn":"arn:aws:iam::123456789012:role/
CWLtoKinesisRole", "DestinationArn":"arn:aws:kinesis:region:123456789012:stream/
TestStream", "FilterPattern": "Test", "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

8. 設定訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式和選取條件的所有傳入日誌事件轉送到您的串流。

   欄位是選用selection-criteria的，但對於排除可能導致訂閱篩選條件無限日誌遞迴的日誌群組非常重要。如需此問題和決定要排除哪些日誌群組的詳細資訊，請參閱 日誌遞迴預防。目前，NOT IN 是 唯一支援的運算子selection-criteria。

   您可以使用 Amazon Kinesis Data Streams 碎片迭代器和使用 Amazon Kinesis Data Streams get-records命令擷取一些 Amazon Kinesis Data Streams 記錄，來驗證日誌事件的流程：

```
aws kinesis get-shard-iterator --stream-name TestStream --shard-id
 shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
```

在 Amazon Kinesis Data Streams 開始傳回資料之前，您可能需要使用此命令幾次。

您應該預期會看到含一系列的記錄的回應。Amazon Kinesis Data Streams 記錄中的資料屬性是 base64 編碼並以 gzip 格式壓縮。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "messageType": "DATA_MESSAGE",
    "owner": "123456789012",
    "logGroup": "Example1",
    "logStream": "logStream1",
    "subscriptionFilters": [
        "ExamplePolicy"
    ],
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424298221568",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221569",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221570",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
```

```
        }
    ],
    "policyLevel": "ACCOUNT_LEVEL_POLICY"
}
```

資料結構中的關鍵元素如下：

messageType

資料訊息將使用「DATA_MESSAGE」類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Amazon Kinesis Data Streams 記錄，主要用於檢查目的地是否可連線。

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。「id」屬性是每個記錄事件的唯一識別符。

policyLevel

強制執行政策的層級。"ACCOUNT_LEVEL_POLICY" 是帳戶層級訂閱篩選條件政策`policyLevel`的 。

範例 2：使用 的訂閱篩選條件 AWS Lambda

在此範例中，您將建立 CloudWatch Logs 帳戶層級訂閱篩選條件政策，將日誌資料傳送至您的 AWS Lambda 函數。

> **Warning**
>
> 建立 Lambda 函數前，請計算將產生的日誌資料量。請務必建立可以處理此磁碟區的函數。如果函數無法處理磁碟區，則會調節日誌串流。由於所有日誌群組或帳戶日誌群組子集的日誌事件都會轉送至目的地，因此會有限流風險。如需 Lambda 限制的詳細資訊，請參閱 AWS Lambda 限制。

為 Lambda 建立帳戶層級訂閱篩選條件政策

1. 建立 AWS Lambda 函數。

   確保您已設定 Lambda 執行角色。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的步驟 2.2：建立 IAM 角色 (執行角色)。
2. 開啟文字編輯器，並建立名為 `helloWorld.js` 的檔案，內含下列內容：

```
var zlib = require('zlib');
exports.handler = function(input, context) {
    var payload = Buffer.from(input.awslogs.data, 'base64');
    zlib.gunzip(payload, function(e, result) {
        if (e) {
            context.fail(e);
        } else {
            result = JSON.parse(result.toString());
            console.log("Event Data:", JSON.stringify(result, null, 2));
            context.succeed();
        }
    });
};
```

3. 壓縮檔案 helloWorld.js，並以名稱 `helloWorld.zip` 將其儲存。
4. 使用下列命令，其中角色是您在第一個步驟中設定的 Lambda 執行角色：

```
aws lambda create-function \
    --function-name helloworld \
    --zip-file fileb://file-path/helloWorld.zip \
    --role lambda-execution-role-arn \
    --handler helloWorld.handler \
    --runtime nodejs18.x
```

5. 授予 CloudWatch Logs 許可來執行函數。使用下列命令，將預留位置帳戶取代為您自己的帳戶。

```
aws lambda add-permission \
    --function-name "helloworld" \
    --statement-id "helloworld" \
    --principal "logs.amazonaws.com" \
    --action "lambda:InvokeFunction" \
    --source-arn "arn:aws:logs:region:123456789012:log-group:*" \
    --source-account "123456789012"
```

6. 使用下列命令建立帳戶層級訂閱篩選條件政策，將預留位置帳戶取代為您自己的帳戶。在此範例中，所有包含字串的日誌事件ERROR都會串流，但名為 LogGroupToExclude1和 的日誌群組中的日誌事件除外LogGroupToExclude2。

```
aws logs put-account-policy \
    --policy-name "ExamplePolicyLambda" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document
 '{"DestinationArn":"arn:aws:lambda:region:123456789012:function:helloWorld",
 "FilterPattern": "Test", "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

設定訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式和選取條件的所有傳入日誌事件轉送到您的串流。

欄位是選用selection-criteria的，但對於排除可能導致訂閱篩選條件無限日誌遞迴的日誌群組非常重要。如需此問題和決定要排除哪些日誌群組的詳細資訊，請參閱 日誌遞迴預防。目前，NOT IN 是 唯一支援的運算子selection-criteria。

7. (選用) 使用範例日誌事件進行測試。在命令提示字元中執行下列命令，這會將簡單日誌訊息放置到訂閱的串流。

若要查看 Lambda 函數的輸出，請導覽至 Lambda 函數，其中您將會在 /aws/lambda/helloworld 中看到輸出：

```
aws logs put-log-events --log-group-name Example1 --log-stream-name logStream1 --
log-events "[{\"timestamp\":CURRENT TIMESTAMP MILLIS , \"message\": \"Simple Lambda
 Test\"}]"
```

預期會看到含一系列 Lambda 的回應。Lambda 記錄中的 Data (資料) 屬性是以 Base64 編碼並以 gzip 格式壓縮。Lambda 收到的實際酬載為以下格式：`{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }`。您可以從命令列使用以下 Unix 命令來檢視原始資料：

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "messageType": "DATA_MESSAGE",
    "owner": "123456789012",
    "logGroup": "Example1",
    "logStream": "logStream1",
    "subscriptionFilters": [
        "ExamplePolicyLambda"
    ],
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424298221568",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221569",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424298221570",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\"Root\"}"
        }
    ],
    "policyLevel": "ACCOUNT_LEVEL_POLICY"
}
```

> **Note**
>
> 帳戶層級訂閱篩選條件不會套用至目的地 Lambda 函數的日誌群組。這是為了防止無限日誌遞迴，這可能會導致擷取帳單增加。如需此問題的詳細資訊，請參閱 日誌遞迴預防 。

資料結構中的關鍵元素如下：

messageType

資料訊息將使用「DATA_MESSAGE」類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Amazon Kinesis Data Streams 記錄，主要用於檢查目的地是否可連線。

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。「id」屬性是每個記錄事件的唯一識別符。

policyLevel

強制執行政策的層級。"ACCOUNT_LEVEL_POLICY" 是帳戶層級訂閱篩選條件政策`policyLevel`的 。

範例 3：使用 Amazon Data Firehose 的訂閱篩選條件

在此範例中，您將建立 CloudWatch Logs 帳戶層級訂閱篩選條件政策，將符合您所定義篩選條件的傳入日誌事件傳送至 Amazon Data Firehose 交付串流。從 CloudWatch Logs 傳送至 Amazon

Data Firehose 的資料已使用 gzip 第 6 級壓縮進行壓縮，因此您不需要在 Firehose 交付串流中使用壓縮。然後，您可以使用 Firehose 中的解壓縮功能自動解壓縮日誌。如需詳細資訊，請參閱使用 CloudWatch Logs 寫入 Kinesis Data Firehose。

> **Warning**
>
> 建立 Firehose 串流之前，請先計算將產生的日誌資料量。請務必建立可處理此磁碟區的 Firehose 串流。如果串流無法處理磁碟區、日誌串流將受到限制。如需 Firehose 串流磁碟區限制的詳細資訊，請參閱 Amazon Data Firehose Data Limits。

建立 Firehose 的訂閱篩選條件

1. 建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體。我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，請跳到步驟 2。

   執行以下命令，將預留位置 Region 換成您想要使用的區域：

   ```
   aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration
    LocationConstraint=region
   ```

   下列為範例輸出：

   ```
   {
       "Location": "/amzn-s3-demo-bucket2"
   }
   ```

2. 建立 IAM 角色，授予 Amazon Data Firehose 將資料放入 Amazon S3 儲存貯體的許可。

   如需詳細資訊，請參閱《Amazon Data Firehose 開發人員指南》中的使用 Amazon Data Firehose 控制存取。

   首先，請按如下所示，使用文字編輯器來建立檔案 ~/TrustPolicyForFirehose.json 中的信任政策：

   ```
   {
     "Statement": {
       "Effect": "Allow",
       "Principal": { "Service": "firehose.amazonaws.com" },
       "Action": "sts:AssumeRole"
   ```

```
    }
}
```

3. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請記下傳回的 Role.Arn 值，因為在後續步驟中會需要它：

```
aws iam create-role \
 --role-name FirehosetoS3Role \
 --assume-role-policy-document file://~/TrustPolicyForFirehose.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "firehose.amazonaws.com"
                }
            }
        },
        "RoleId": "EXAMPLE50GAB4HC5F431",
        "CreateDate": "2023-05-29T13:46:29.431Z",
        "RoleName": "FirehosetoS3Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/FirehosetoS3Role"
    }
}
```

4. 建立許可政策，以定義 Firehose 可以在您的帳戶上執行的動作。首先，使用文字編輯器來建立檔案 ~/PermissionsForFirehose.json 中的許可政策：

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "s3:AbortMultipartUpload",
          "s3:GetBucketLocation",
          "s3:GetObject",
          "s3:ListBucket",
          "s3:ListBucketMultipartUploads",
          "s3:PutObject" ],
```

```
      "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket2",
          "arn:aws:s3:::amzn-s3-demo-bucket2/*" ]
    }
  ]
}
```

5. 使用以下 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-
Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. 如下所示建立目的地 Firehose 交付串流，將 RoleARN 和 BucketARN 的預留位置值取代為您建立的角色和儲存貯體 ARNs：

```
aws firehose create-delivery-stream \
   --delivery-stream-name 'my-delivery-stream' \
   --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::123456789012:role/FirehosetoS3Role", "BucketARN":
 "arn:aws:s3:::amzn-s3-demo-bucket2"}'
```

NFirehose 會自動為交付的 Amazon S3 物件使用 YYYY/MM/DD/HH UTC 時間格式的字首。您可以指定在時間格式前綴前要新增的額外前綴。如果字首結尾是斜線 (/)，則會在 Amazon S3 儲存貯體中顯示為資料夾。

7. 等待幾分鐘，讓串流變成作用中。您可以使用 Firehose describe-delivery-stream 命令來檢查 DeliveryStreamDescription.DeliveryStreamStatus 屬性。此外，請留意 DeliveryStreamDescription.DeliveryStreamARN 值，因您將在後續步驟需要此值：

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
{
    "DeliveryStreamDescription": {
        "HasMoreDestinations": false,
        "VersionId": "1",
        "CreateTimestamp": 1446075815.822,
        "DeliveryStreamARN": "arn:aws:firehose:us-
east-1:123456789012:deliverystream/my-delivery-stream",
        "DeliveryStreamStatus": "ACTIVE",
        "DeliveryStreamName": "my-delivery-stream",
        "Destinations": [
            {
```

```
                "DestinationId": "destinationId-000000000001",
                "S3DestinationDescription": {
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "RoleARN": "delivery-stream-role",
                    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket2",
                    "BufferingHints": {
                        "IntervalInSeconds": 300,
                        "SizeInMBs": 5
                    }
                }
            }
        ]
    }
}
```

8. 建立 IAM 角色，授予 CloudWatch Logs 將資料放入 Firehose 交付串流的許可。首先，使用文字編輯器來建立檔案 ~/TrustPolicyForCWL.json 中的信任政策：

   此政策包含 aws:SourceArn 全域條件內容金鑰，以協助預防混淆代理人安全問題。如需詳細資訊，請參閱預防混淆代理人。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
         "StringLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
     }
  }
}
```

9. 使用 create-role 命令來建立 IAM 角色，並指定信任政策檔案。請記下傳回的 Role.Arn 值，因為在後續步驟中會需要它：

```
aws iam create-role \
--role-name CWLtoKinesisFirehoseRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json
```

```
{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Condition": {
                     "StringLike": {
                         "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
                     }
                 }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
    }
}
```

10. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器來建立許可政策檔案 (例如，~/PermissionsForCWL.json)：

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
      }
    ]
}
```

11. 使用 put-role-policy 命令將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. 在 Amazon Data Firehose 交付串流處於作用中狀態且您已建立 IAM 角色後，您可以建立 CloudWatch Logs 帳戶層級訂閱篩選條件政策。此政策會立即啟動從所選日誌群組到 Amazon Data Firehose 交付串流的即時日誌資料流程：

```
aws logs put-account-policy \
    --policy-name "ExamplePolicyFirehose" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document '{"RoleArn":"arn:aws:iam::123456789012:role/
CWLtoKinesisFirehoseRole", "DestinationArn":"arn:aws:firehose:us-
east-1:123456789012:deliverystream/delivery-stream-name", "FilterPattern": "Test",
 "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

13. 設定訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式的傳入日誌事件轉送至您的 Amazon Data Firehose 交付串流。

    欄位是選用`selection-criteria`的，但對於排除可能導致訂閱篩選條件無限日誌遞迴的日誌群組非常重要。如需此問題和決定要排除哪些日誌群組的詳細資訊，請參閱 日誌遞迴預防。目前，NOT IN 是 唯一支援的運算子`selection-criteria`。

    您的資料會根據 Amazon Data Firehose 交付串流上設定的時間緩衝間隔，開始出現在 Amazon S3 中。一旦經過足夠的時間，您就可以檢查 Amazon S3 儲存貯體來驗證資料。

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket2' --prefix 'firehose/'
{
    "Contents": [
        {
            "LastModified": "2023-10-29T00:01:25.000Z",
            "ETag": "\"a14589f8897f4089d3264d9e2d1f1610\"",
            "StorageClass": "STANDARD",
            "Key": "firehose/2015/10/29/00/my-delivery-stream-2015-10-29-00-01-21-
a188030a-62d2-49e6-b7c2-b11f1a7ba250",
            "Owner": {
                "DisplayName": "cloudwatch-logs",
                "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b5"
            },
```

```
            "Size": 593
        },
        {
            "LastModified": "2015-10-29T00:35:41.000Z",
            "ETag": "\"a7035b65872bb2161388ffb63dd1aec5\"",
            "StorageClass": "STANDARD",
            "Key": "firehose/2023/10/29/00/my-delivery-stream-2023-10-29-00-35-40-
EXAMPLE-7e66-49bc-9fd4-fc9819cc8ed3",
            "Owner": {
                "DisplayName": "cloudwatch-logs",
                "ID": "EXAMPLE6be062b19584e0b7d84ecc19237f87b6"
            },
            "Size": 5752
        }
    ]
}
```

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket2' --key 'firehose/2023/10/29/00/
my-delivery-stream-2023-10-29-00-01-21-a188030a-62d2-49e6-b7c2-b11f1a7ba250'
 testfile.gz

{
    "AcceptRanges": "bytes",
    "ContentType": "application/octet-stream",
    "LastModified": "Thu, 29 Oct 2023 00:07:06 GMT",
    "ContentLength": 593,
    "Metadata": {}
}
```

在 Amazon S3 物件中的資料會以 gzip 格式壓縮。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
zcat testfile.gz
```

跨帳戶跨區域訂閱

您可以與不同 AWS 帳戶的擁有者合作，並在您的 AWS 資源上接收其日誌事件，例如 Amazon Kinesis 或 Amazon Data Firehose 串流（這稱為跨帳戶資料共用）。例如，您可以從集中式 Amazon Kinesis Data Streams 或 Firehose 串流讀取此日誌事件資料，以執行自訂處理和分析。自訂處理在您進行跨多個帳戶的協作和分析資料時特別有用。

例如，公司的資訊安全群組可能要分析即時入侵偵測或異常行為的資料，讓它可以對公司所有部門的帳戶進行稽核，方法是收集他們的聯合身分生產日誌以集中處理。跨這些帳戶的事件資料即時串流可以組合並交付給資訊安全群組，而資訊安全群組可以使用 Amazon Kinesis Data Streams 將資料連接到其現有的安全分析系統。

> **Note**
>
> 日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地指向的 AWS 資源可以位於不同的區域。在下列各節中的範例中，所有區域特定的資源都會在美國東部（維吉尼亞北部）建立）。

如果您已設定 AWS Organizations 並正在使用成員帳戶，您可以使用日誌集中，從來源帳戶收集日誌資料到中央監控帳戶。

使用集中式日誌群組時，您可以在建立訂閱篩選條件時使用下列系統欄位維度：

- `@aws.account` - 此維度代表日誌事件來源 AWS 的帳戶 ID。
- `@aws.region` - 此維度代表產生日誌事件 AWS 的區域。

這些維度有助於識別日誌資料來源，以便更精細地篩選和分析衍生自集中式日誌的指標。

主題

使用 Amazon Kinesis Data Streams 的跨帳戶跨區域日誌資料共用

建立跨帳戶訂閱時，您可以指定單一帳戶或一個組織作為寄件者。如果您指定組織，則此程序會讓組織中的所有帳戶都能將日誌傳送至接收者帳戶。

若要跨帳戶共用日誌資料，您需要建立日誌資料寄件者和接收者：

- Log data sender (日誌資料寄件者) - 從收件人取得目的地資訊，並讓 CloudWatch Logs 知道可開始將日誌事件傳送到指定的目的地。在本節其餘部分的程序中，日誌資料寄件者顯示的虛構 AWS 帳戶號碼為 111111111111。

  如果您要讓一個組織中的多個帳戶將日誌傳送至一個收件人帳戶，則可以建立一個政策，授予組織中所有帳戶將日誌傳送至收件人帳戶的許可。您仍然必須為每個寄件者帳戶設定個別的訂閱篩選條件。
- 日誌資料收件人 - 設定封裝 Amazon Kinesis Data Streams 串流的目的地，並讓 CloudWatch Logs 知道收件人想要接收日誌資料。然後，收件人接著會與寄件者共用與其目的地有關的資訊。在本節其餘部分的程序中，日誌資料收件人的虛構 AWS 帳戶號碼為 999999999999。

若要從跨帳戶使用者開始接收日誌事件，日誌資料收件人會先建立 CloudWatch Logs 目的地。每個目的地包含以下關鍵元素：

目的地名稱

您要建立的目的地名稱。

目標 ARN

您要用作訂閱摘要目的地之 AWS 資源的 Amazon Resource Name (ARN)。

角色 ARN

授予 CloudWatch Logs 將資料放入所選串流的必要許可的 AWS Identity and Access Management (IAM) 角色。

存取政策

IAM 政策文件 (JSON 格式，使用 IAM 政策文法撰寫)，決定允許一組使用者寫入您的目的地。

> **Note**
>
> 日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地指向 AWS 的資源可以位於不同的 區域。在以下各節的範例中，區域特定的所有資源都在美國東部 (維吉尼亞北部) 建立。

主題

設定新的跨帳戶訂閱

按照這些章節中的步驟設定新的跨帳戶日誌訂閱。

主題

步驟 1：建立目的地

> **Important**
>
> 此程序中的所有步驟必須在日誌資料收件人帳戶中完成。

在此範例中，日誌資料收件人帳戶的帳戶 AWS ID 為 999999999999，而日誌資料寄件者 AWS 帳戶 ID 為 111111111111。

此範例使用名為 RecipientStream 的 Amazon Kinesis Data Streams 串流建立目的地，以及可讓 CloudWatch Logs 將資料寫入其中的角色。

建立目的地時，CloudWatch Logs 會以收件人帳戶的名義向目的地傳送測試訊息。當訂閱篩選條件在稍後處於作用中狀態時，CloudWatch Logs 會以來源帳戶的名義向目的地傳送日誌事件。

若要建立目的地

1. 在收件人帳戶中，在 Amazon Kinesis Data Streams 中建立目的地串流。在命令提示字元中輸入：

   ```
   aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
   ```

2. 等到 串流變成作用中。您可以使用以下 aws kinesis describe-stream 命令來檢查 StreamDescription.StreamStatus 屬性。此外，應記下 StreamDescription.StreamARN 值，因為稍後要將該值傳遞至 CloudWatch Logs：

```
aws kinesis describe-stream --stream-name "RecipientStream"
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "RecipientStream",
    "StreamARN": "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "340282366920938463463374607431768EXAMPLE",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
 "49551135218688818456679503831981458784591352702 18EXAMPLE"
        }
      }
    ]
  }
}
```

這可能需要花費幾分鐘，讓串流以作用中狀態出現。

3. 建立將授予 CloudWatch Logs 許可以將資料放到串流中的 IAM 角色。首先，您將需要在檔案 ~/TrustPolicyForCWL.json 中建立信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

   此政策包含 aws:SourceArn 全域條件內容金鑰，可指定 sourceAccountId 以協助預防混淆代理人安全問題。如果您在第一次呼叫中還不知道來源帳戶 ID，我們建議您將目的地 ARN 放在來源 ARN 欄位中。在後續呼叫中，應將來源 ARN 設定為從第一次呼叫中收集的實際來源 ARN。如需詳細資訊，請參閱預防混淆代理人。

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Condition": {
            "StringLike": {
```

```
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
}
```

4. 使用 aws iam create-role 命令來建立 IAM 角色，並指定信任政策檔案。記下傳回的 Role.Arn 值，因為稍後也要將此值傳遞至 CloudWatch Logs：

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                },
                "Principal": {
                    "Service": "logs.amazonaws.com"
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器在檔案 ~/PermissionsForCWL.json 中建立許可政策：

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. 使用 aws iam put-role-policy 命令將許可政策與角色相關聯：

```
aws iam put-role-policy \
    --role-name CWLtoKinesisRole \
    --policy-name Permissions-Policy-For-CWL \
    --policy-document file://~/PermissionsForCWL.json
```

7. 在串流處於作用中狀態且您已建立 IAM 角色後，即可建立 CloudWatch Logs 目的地。

   a. 此步驟不會將存取政策與您的目的地相關聯，且只是完成目的地建立兩步驟中的第一步。請注意在承載中傳回的 DestinationArn：

```
aws logs put-destination \
    --destination-name "testDestination" \
    --target-arn "arn:aws:kinesis:region:999999999999:stream/RecipientStream" \
    --role-arn "arn:aws:iam::999999999999:role/CWLtoKinesisRole"

{
  "DestinationName" : "testDestination",
  "RoleArn" : "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
  "DestinationArn" : "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
  "TargetArn" : "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream"
}
```

   b. 步驟 7a 完成後，即可在日誌資料收件人帳戶中，將存取政策與目的地建立關聯。此政策必須指定 logs:PutSubscriptionFilter 動作，並授予寄件者帳戶許可以存取目的地。

政策會將許可授予傳送日誌 AWS 的帳戶。您可以在政策中僅指定這一個帳戶，或者如果寄件者帳戶是組織的成員，則政策可以指定組織的組織 ID。如此一來，您可以僅建立一個政策，就能允許一個組織中的多個帳戶將日誌傳送至此目的地帳戶。

使用文字編輯器建立名為 ~/AccessPolicy.json 的檔案，並隨附下列其中一個政策陳述。

此第一個範例政策允許組織中具有 ID 為 o-1234567890 的所有帳戶將日誌傳送至收件人帳戶。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-1234567890"
                    ]
                }
            }
        }
    ]
}
```

此下一個範例只允許日誌資料寄件者帳戶 (111111111111) 將日誌傳送至日誌資料收件人帳戶。

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111111111111"
            },
            "Action": "logs:PutSubscriptionFilter",
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination"
        }
    ]
}
```

c. 將您在上一步驟建立的政策連接到目的地。

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

此存取政策可讓 ID 為 111111111111 的 AWS 帳戶中的使用者針對 ARN arn：aws：logs：*region*：999999999999：destination：testDestination 的目的地呼叫 PutSubscriptionFilter。任何其他使用者針對此目的地呼叫 PutSubscriptionFilter 的嘗試將遭到拒絕。

若要驗證使用者的權限符合存取政策，請參閱《IAM 使用者指南》中的使用政策驗證程式。

完成後，如果您將 AWS Organizations 用於跨帳戶許可，請遵循中的步驟步驟 2：(僅限於使用組織時) 建立 IAM 角色。如果您要將許可直接授予給其他帳戶，而不是使用 Organizations，則可以略過該步驟並繼續進行 步驟 4：建立訂閱篩選條件。

步驟 2：(僅限於使用組織時) 建立 IAM 角色

在上一節中，如果您藉由使用授予許可給帳戶 111111111111 所屬組織的存取政策來建立目的地，而不是將許可直接授予給帳戶 111111111111，則按照本節中的步驟進行。若否，則可跳至步驟 步驟 4：建立訂閱篩選條件。

本節中的步驟會建立 IAM 角色，CloudWatch 可以代入該角色並驗證寄件者帳戶是否具有針對收件人目的地建立訂閱篩選條件的許可。

在寄件者帳戶中執行此區段中的步驟。角色必須存在於寄件者帳戶中，而且您要在訂閱篩選條件中指定此角色的 ARN。在此範例中，使用者帳戶為 111111111111。

使用 建立跨帳戶日誌訂閱所需的 IAM 角色 AWS Organizations

1. 在檔案 /TrustPolicyForCWLSubscriptionFilter.json 中建立下列信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 建立使用此政策的 IAM 角色。記下命令傳回的 Arn 值，之後在此程序中會用到。在此範例中，我們使用 CWLtoSubscriptionFilterRole 作為要建立的角色的名稱。

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行的動作。

   a. 首先，使用文字編輯器在名為 ~/PermissionsForCWLSubscriptionFilter.json 的檔案中建立下列許可政策。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 輸入下列命令，將您剛建立的許可政策與您在步驟 2 中建立的角色相關聯。

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

完成後，可以繼續進行 步驟 4：建立訂閱篩選條件。

步驟 3：新增/驗證跨帳戶目的地的 IAM 許可

根據 AWS 跨帳戶政策評估邏輯，若要存取任何跨帳戶資源 （例如做為訂閱篩選條件目的地的 Kinesis 或 Firehose 串流），您必須在傳送帳戶中擁有身分型政策，以明確存取跨帳戶目的地資源。如需有關政策評估邏輯的詳細資訊，請參閱《跨帳戶政策評估邏輯》。

您可以將身分型政策連接至用來建立訂閱篩選條件的 IAM 角色或 IAM 使用者。傳送帳戶中必須有此政策存在。如果您使用管理員角色建立訂閱篩選條件，則可以略過此步驟並繼續進行 步驟 4：建立訂閱篩選條件。

新增或驗證跨帳戶所需的 IAM 許可

1. 輸入下列命令，以檢查要用來執行 AWS 日誌命令的 IAM 角色或 IAM 使用者。

   ```
   aws sts get-caller-identity
   ```

   此命令會傳回類似以下的輸出：

   ```
   {
   "UserId": "User ID",
   "Account": "sending account id",
   "Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
   }
   ```

   記下以 *RoleName* 或 *UserName* 表示的值。
2. 登入傳送帳戶中 AWS 管理主控台 的 ，並使用您在步驟 1 中輸入的命令輸出中傳回的 IAM 角色或 IAM 使用者來搜尋附加的政策。
3. 確認連接至此角色或使用者的政策提供明確的許可，可對跨帳戶目的地資源呼叫 `logs:PutSubscriptionFilter`。

下列政策提供僅在單一 AWS 帳戶 中在任何目的地資源上建立訂閱篩選條件的許可999999999999：

JSON

```
{
     "Version":"2012-10-17",
     "Statement": [
        {
            "Sid": "AllowSubscriptionFiltersOnAccountResources",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:*"
            ]
        }
    ]
}
```

下列政策提供僅在單一 AWS 帳戶 sampleDestination中名為 的特定目的地資源上建立訂閱篩選條件的許可123456789012：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSubscriptionFiltersonAccountResource",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:sampleDestination"
            ]
        }
    ]
```

```
}
```

步驟 4：建立訂閱篩選條件

在建立目的地後，日誌資料收件人帳戶可以與其他 AWS 帳戶共用目的地 ARN (arn:aws:logs:us-east-1:999999999999:destination:testDestination)，讓他們能將日誌事件傳送到相同目的地。然後，這些其他傳送帳戶使用者接著會在個別的日誌群組針對此目的地建立訂閱篩選條件。訂閱篩選條件會立即開始將即時日誌資料從所選的日誌群組傳送到指定的目標。

> **Note**
>
> 如果您要將訂閱篩選條件的許可授予給整個組織，您需要使用您在 步驟 2：(僅限於使用組織時) 建立 IAM 角色 中建立的 IAM 角色 ARN。

在下列範例中，訂閱篩選條件是在傳送帳戶中建立的。篩選條件與包含 AWS CloudTrail 事件的日誌群組相關聯，因此「根」 AWS 憑證所做的每個記錄活動都會傳送到您先前建立的目的地。該目標會封裝名為「RecipientStream」的 串流。

下列部分中的其餘步驟，假定您已遵照將 CloudTrail 事件傳送至 CloudWatch Logs 中的 AWS CloudTrail 使用者指南，並建立了一個包含您的 CloudTrail 事件的日誌群組。這些步驟假定此日誌群組的名稱為 `CloudTrail/logs`。

當您輸入以下命令時，確認您以 IAM 使用者身分登入，或是使用您在 步驟 3：新增/驗證跨帳戶目的地的 IAM 許可 中為其新增政策的 IAM 角色登入。

```
aws logs put-subscription-filter \
    --log-group-name "CloudTrail/logs" \
    --filter-name "RecipientStream" \
    --filter-pattern "{$.userIdentity.type = Root}" \
    --destination-arn "arn:aws:logs:region:999999999999:destination:testDestination"
```

日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地可以指向 AWS 資源，例如位於不同區域的 Amazon Kinesis Data Streams 串流。

驗證日誌事件的流動

在您建立訂閱篩選條件後，CloudWatch Logs 會將所有符合篩選條件模式的傳入日誌事件，轉送至封裝在稱為 "RecipientStream" 的目的地串流中的串流。目的地擁有者可以使用 aws kinesis get-shard-

iterator 命令來抓取 Amazon Kinesis Data Streams 碎片，並使用 aws kinesis get-records 命令來擷取一些 Amazon Kinesis Data Streams 記錄，以確認是否發生這種情況：

```
aws kinesis get-shard-iterator \
      --stream-name RecipientStream \
      --shard-id shardId-000000000000 \
      --shard-iterator-type TRIM_HORIZON

{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
}

aws kinesis get-records \
      --limit 10 \
      --shard-iterator
      "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

> **Note**
>
> 在 Amazon Kinesis Data Streams 開始傳回資料之前，您可能需要重新執行 get-records 命令幾次。

您應該會看到具有 Amazon Kinesis Data Streams 記錄陣列的回應。Amazon Kinesis Data Streams 記錄中的資料屬性會以 gzip 格式壓縮，然後以 base64 編碼。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "owner": "111111111111",
```

```
    "logGroup": "CloudTrail/logs",
    "logStream": "111111111111_CloudTrail/logs_us-east-1",
    "subscriptionFilters": [
        "RecipientStream"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        }
    ]
}
```

在資料結構的關鍵元素如下：

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

messageType

資料訊息使用 "DATA_MESSAGE" 類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Amazon Kinesis Data Streams 記錄，主要用於檢查目的地是否可連線。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。ID 屬性是每個記錄事件的唯一識別符。

在執行期修改目的地成員資格

您可能遇到以下情況：您需要從擁有的目的地中新增或移除某些使用者的成員資格。您可以在目的地使用 `put-destination-policy` 命令並指定新的存取政策。在下列範例中，之前新增帳戶 111111111111 會停止傳送任何更多資料，且帳戶 222222222222 會啟用。

1. 擷取目前與目的地 testDestination 相關聯的政策，並留意 AccessPolicy：

```
aws logs describe-destinations \
    --destination-name-prefix "testDestination"

{
 "Destinations": [
   {
     "DestinationName": "testDestination",
     "RoleArn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
     "DestinationArn":
 "arn:aws:logs:region:999999999999:destination:testDestination",
     "TargetArn": "arn:aws:kinesis:region:999999999999:stream/RecipientStream",
     "AccessPolicy": "{\"Version\": \"2012-10-17\", \"Statement\":
 [{\"Sid\": \"\", \"Effect\": \"Allow\", \"Principal\": {\"AWS\":
 \"111111111111\"}, \"Action\": \"logs:PutSubscriptionFilter\", \"Resource\":
 \"arn:aws:logs:region:999999999999:destination:testDestination\"}] }"
   }
 ]
}
```

2. 更新政策，以反映帳戶 111111111111 已停用，且帳戶 222222222222 已啟用。將此政策放置在 ~/NewAccessPolicy.json 檔案中：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "222222222222"
            },
            "Action": "logs:PutSubscriptionFilter",
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination"
        }
    ]
}
```

3. 呼叫 PutDestinationPolicy 以與含目的地之 NewAccessPolicy.json 檔案中定義的政策相關聯：

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

這最終會從帳戶 ID 111111111111 停用日誌事件。在帳戶 222222222222 的擁有者建立訂閱篩選條件後，來自帳戶 ID 222222222222 的日誌事件會立刻開始流向目的地。

更新現有的跨帳戶訂閱

如果您目前有跨帳戶日誌訂閱，其中目的地帳戶僅授予特定寄件者帳戶許可，而您想要更新此訂閱，讓目的地帳戶授予組織中所有帳戶的存取權，請按照本節中的步驟進行。

主題

- 步驟 1：更新訂閱篩選條件
- 步驟 2：更新現有的目的地存取政策

步驟 1：更新訂閱篩選條件

> **Note**
>
> 只有跨帳戶訂閱由 從 AWS 服務啟用記錄 所列服務建立的日誌才需要此步驟。如果您不使用這些日誌群組之一建立的日誌，則可以跳至 步驟 2：更新現有的目的地存取政策。

在某些情況下，您必須更新所有傳送日誌至目的地帳戶的寄件者帳戶中的訂閱篩選條件。此更新會新增 IAM 角色，CloudWatch 可以代入該角色並驗證寄件者帳戶是否具有將日誌傳送至收件人帳戶的許可。

針對您想要更新的每個寄件者帳戶，按照本節中的步驟進行，以將組織 ID 用於跨帳戶訂閱許可。

在本節的範例中，111111111111 和 222222222222 兩個帳戶已經建立訂閱篩選條件，以將日誌傳送至帳戶 999999999999。現有的訂閱篩選條件值如下：

```
## Existing Subscription Filter parameter values
    \ --log-group-name "my-log-group-name"
    \ --filter-name "RecipientStream"
    \ --filter-pattern "{$.userIdentity.type = Root}"
    \ --destination-arn "arn:aws:logs:region:999999999999:destination:testDestination"
```

如果需要尋找目前的訂閱篩選條件參數值，請輸入下列命令。

```
aws logs describe-subscription-filters
    \ --log-group-name "my-log-group-name"
```

更新訂閱篩選條件以開始將組織 ID 用於跨帳戶日誌許可

1. 在檔案 ~/TrustPolicyForCWL.json 中建立下列信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 建立使用此政策的 IAM 角色。記下命令傳回的 Arn 值的 Arn 值，之後在此程序中會用到。在此範例中，我們使用 CWLtoSubscriptionFilterRole 作為要建立的角色的名稱。

```
aws iam create-role
    \ --role-name CWLtoSubscriptionFilterRole
    \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行的動作。

   a. 首先，使用文字編輯器在名為 /PermissionsForCWLSubscriptionFilter.json 的檔案中建立下列許可政策。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 輸入下列命令，將您剛建立的許可政策與您在步驟 2 中建立的角色相關聯。

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. 輸入下列命令以更新訂閱篩選條件。

```
aws logs put-subscription-filter
    \ --log-group-name "my-log-group-name"
    \ --filter-name "RecipientStream"
    \ --filter-pattern "{$.userIdentity.type = Root}"
    \ --destination-arn
 "arn:aws:logs:region:999999999999:destination:testDestination"
    \ --role-arn "arn:aws:iam::111111111111:role/CWLtoSubscriptionFilterRole"
```

步驟 2：更新現有的目的地存取政策

更新所有寄件者帳戶中的訂閱篩選條件之後，您可以更新收件人帳戶中的目的地存取政策。

下列範例中，收件人帳戶為 999999999999，且目的地名稱為 testDestination。

此更新可讓屬於組織且 ID 為 o-1234567890 的所有帳戶傳送日誌至收件人帳戶。只有已建立訂閱篩選條件的帳戶才會真的傳送日誌至收件人帳戶。

更新收件人帳戶中的目的地存取政策，以開始將組織 ID 用於許可

1. 在收件人帳戶中，使用文字編輯器建立 ~/AccessPolicy.json 檔案，其中包含以下內容。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-1234567890"
                    ]
                }
            }
        }
    ]
}
```

2. 輸入下列命令，將您剛建立的政策連接到現有的目的地。若要更新目的地以使用具有組織 ID 的存取政策，而非列出特定 AWS 帳戶 IDs的存取政策，請包含 force 參數。

> **Warning**
>
> 如果您使用 中列出的 AWS 服務傳送的日誌從 AWS 服務啟用記錄，則在執行此步驟之前，您必須先更新所有寄件者帳戶中的訂閱篩選條件，如 中所述步驟 1：更新訂閱篩選條件。

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
    \ --force
```

使用 Firehose 的跨帳戶跨區域日誌資料共用

若要跨帳戶共用日誌資料，您需要建立日誌資料寄件者和接收者：

- 日誌資料寄件者 - 從收件人取得目的地資訊，並讓 CloudWatch Logs 知道可開始將日誌事件傳送到指定的目的地。在本節其餘部分的程序中，日誌資料寄件者顯示的虛構 AWS 帳戶號碼為 111111111111。
- 日誌資料收件人 - 設定封裝 Amazon Kinesis Data Streams 串流的目的地，並讓 CloudWatch Logs 知道收件人想要接收日誌資料。然後，收件人接著會與寄件者共用與其目的地有關的資訊。在本節其餘部分的程序中，日誌資料收件人的虛擬 AWS 帳戶號碼為 222222222222。

本節中的範例使用 Firehose 交付串流搭配 Amazon S3 儲存。您也可以使用不同的設定來設定 Firehose 交付串流。如需詳細資訊，請參閱建立 Firehose 交付串流。

> **Note**
>
> 日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地指向 AWS 的資源可以位於不同的 區域。

> **Note**
>
> 支援相同帳戶和跨區域交付串流的 Firehose 訂閱篩選條件。

主題

步驟 1：建立 Firehose 交付串流

> **Important**
>
> 在完成下列步驟之前，您必須使用 存取政策，以便 Firehose 可以存取您的 Amazon S3 儲存貯體。如需詳細資訊，請參閱《Amazon Data Firehose 開發人員指南》中的控制存取。
> 必須在日誌資料收件人帳戶中完成本區段 (步驟 1) 中的所有步驟。
> 在以下範例命令中使用美國東部 (維吉尼亞北部)。請將此區域替換成您部署的正確區域。

建立要用作目的地的 Firehose 交付串流

1. 建立 Amazon S3 儲存貯體：

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-1
```

2. 建立 IAM 角色，授予 Firehose 將資料放入儲存貯體的許可。

   a. 首先，使用文字編輯器在檔案 ~/TrustPolicyForFirehose.json 中建立信任政策。

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
 "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
 { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
```

   b. 建立 IAM 角色，並指定您剛建立的信任政策檔案。

```
aws iam create-role \
    --role-name FirehosetoS3Role \
    --assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

c. 此命令的輸出看起來如下：記下角色名稱和角色 ARN。

```
{
    "Role": {
        "Path": "/",
        "RoleName": "FirehosetoS3Role",
        "RoleId": "AROAR3BXASEKW7K635M53",
        "Arn": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
        "CreateDate": "2021-02-02T07:53:10+00:00",
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Effect": "Allow",
                "Principal": {
                    "Service": "firehose.amazonaws.com"
                },
                "Action": "sts:AssumeRole",
                "Condition": {
                    "StringEquals": {
                        "sts:ExternalId": "222222222222"
                    }
                }
            }
        }
    }
}
```

3. 建立許可政策，以定義 Firehose 可在您的帳戶中執行的動作。

a. 首先，使用文字編輯器在名為 ~/PermissionsForFirehose.json 的檔案中建立下列許可政策。根據您的使用案例，可能需要為此檔案新增更多許可。

```
{
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket",
            "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
```

```
    }]
}
```

b. 輸入下列命令，將您剛建立的許可政策與 IAM 角色相關聯。

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. 輸入下列命令來建立 Firehose 交付串流。以部署的正確值取代 *my-role-arn* 和 *amzn-s3-demo-bucket2-arn*。

```
aws firehose create-delivery-stream \
   --delivery-stream-name 'my-delivery-stream' \
   --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role", "BucketARN":
 "arn:aws:s3:::amzn-s3-demo-bucket"}'
```

輸出格式應類似以下內容：

```
{
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

步驟 2：建立目的地

> **Important**
>
> 此程序中的所有步驟必須在日誌資料收件人帳戶中完成。

建立目的地時，CloudWatch Logs 會以收件人帳戶的名義向目的地傳送測試訊息。當訂閱篩選條件在稍後處於作用中狀態時，CloudWatch Logs 會以來源帳戶的名義向目的地傳送日誌事件。

若要建立目的地

1. 等到您在 中建立的 Firehose 串流步驟 1：建立 Firehose 交付串流變成作用中。您可以使用以下命令來檢查 StreamDescription.StreamStatus 屬性。

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
```

此外，請留意 DeliveryStreamDescription.DeliveryStreamARN 值，因後續步驟中需要用到。此命令的範例輸出：

```
{
    "DeliveryStreamDescription": {
        "DeliveryStreamName": "my-delivery-stream",
        "DeliveryStreamARN": "arn:aws:firehose:us-
east-1:222222222222:deliverystream/my-delivery-stream",
        "DeliveryStreamStatus": "ACTIVE",
        "DeliveryStreamEncryptionConfiguration": {
            "Status": "DISABLED"
        },
        "DeliveryStreamType": "DirectPut",
        "VersionId": "1",
        "CreateTimestamp": "2021-02-01T23:59:15.567000-08:00",
        "Destinations": [
            {
                "DestinationId": "destinationId-000000000001",
                "S3DestinationDescription": {
                    "RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
                    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
                    "BufferingHints": {
                        "SizeInMBs": 5,
                        "IntervalInSeconds": 300
                    },
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    }
                },
                "ExtendedS3DestinationDescription": {
                    "RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
                    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
                    "BufferingHints": {
                        "SizeInMBs": 5,
                        "IntervalInSeconds": 300
                    },
```

```
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    },
                    "S3BackupMode": "Disabled"
                }
            }
        ],
        "HasMoreDestinations": false
    }
}
```

可能需要花費幾分鐘，交付串流才會變成作用中狀態。

2. 當交付串流處於作用中狀態時，請建立 IAM 角色，授予 CloudWatch Logs 將資料放入 Firehose 串流的許可。首先，您將需要在檔案 ~/TrustPolicyForCWL.json 中建立信任政策。請使用文字編輯器來建立此政策。如需 CloudWatch Logs 端點的詳細資訊，請參閱 Amazon CloudWatch Logs 端點和配額。

   此政策包含 aws:SourceArn 全域條件內容金鑰，可指定 sourceAccountId 以協助預防混淆代理人安全問題。如果您在第一次呼叫中還不知道來源帳戶 ID，我們建議您將目的地 ARN 放在來源 ARN 欄位中。在後續呼叫中，應將來源 ARN 設定為從第一次呼叫中收集的實際來源 ARN。如需詳細資訊，請參閱預防混淆代理人。

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.region.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
```

```
     }
}
```

3. 使用 aws iam create-role 命令來建立 IAM 角色，並指定您剛建立的信任政策檔案。

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

下列為範例輸出。請留意傳回的 Role.Arn 值，因為後續步驟中需要用到。

```
{
    "Role": {
        "Path": "/",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "RoleId": "AROAR3BXASEKYJYWF243H",
        "Arn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
        "CreateDate": "2021-02-02T08:10:43+00:00",
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.region.amazonaws.com"
                },
                "Action": "sts:AssumeRole",
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器在檔案 ~/PermissionsForCWL.json 中建立許可政策：

```
{
```

```
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
}
```

5. 輸入以下命令，將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Firehose 交付串流處於作用中狀態且您已建立 IAM 角色後，您可以建立 CloudWatch Logs 目的地。

   a. 此步驟不會將存取政策與您的目的地建立關聯，且為完成建立目的地之兩個步驟的僅第一個步驟。記下承載中傳回的新目的地的 ARN，因為您會在後續步驟中使用它作為 `destination.arn`。

```
aws logs put-destination \

    --destination-name "testFirehoseDestination" \
    --target-arn "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-
delivery-stream" \
    --role-arn "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole"

{
    "destination": {
        "destinationName": "testFirehoseDestination",
        "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
        "roleArn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
        "arn": "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"}
}
```

   b. 上一個步驟完成後，請在日誌資料收件人帳戶中 (222222222222)，將存取政策與目的地建立關聯。

此政策可讓日誌資料寄件者帳戶 (111111111111) 只能在日誌資料收件人帳戶 (222222222222) 中存取目的地。您可以使用文字編輯器來將此政策在放在 ~/AccessPolicy.json 檔案中：

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

c. 這會建立可定義誰擁有對目的地的寫入存取權之政策。此政策必須指定 logs:PutSubscriptionFilter 動作來存取目的地。跨帳戶的使用者將使用 PutSubscriptionFilter 動作來將日誌事件傳送到目的地：

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
    --access-policy file://~/AccessPolicy.json
```

步驟 3：新增/驗證跨帳戶目的地的 IAM 許可

根據 AWS 跨帳戶政策評估邏輯，若要存取任何跨帳戶資源 （例如做為訂閱篩選條件目的地的 Kinesis 或 Firehose 串流），您必須在傳送帳戶中擁有身分型政策，以明確存取跨帳戶目的地資源。如需有關政策評估邏輯的詳細資訊，請參閱《跨帳戶政策評估邏輯》。

您可以將身分型政策連接至用來建立訂閱篩選條件的 IAM 角色或 IAM 使用者。傳送帳戶中必須有此政策存在。如果您使用管理員角色建立訂閱篩選條件，則可以略過此步驟並繼續進行 步驟 4：建立訂閱篩選條件。

新增或驗證跨帳戶所需的 IAM 許可

1. 輸入下列命令，以檢查要用來執行 AWS 日誌命令的 IAM 角色或 IAM 使用者。

```
aws sts get-caller-identity
```

此命令會傳回類似以下的輸出：

```
{
"UserId": "User ID",
"Account": "sending account id",
"Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

記下以 *RoleName* 或 *UserName* 表示的值。

2. 登入傳送帳戶中 AWS 管理主控台 的 ，並使用您在步驟 1 中輸入的命令輸出中傳回的 IAM 角色或 IAM 使用者來搜尋附加的政策。
3. 確認連接至此角色或使用者的政策提供明確的許可，可對跨帳戶目的地資源呼叫 `logs:PutSubscriptionFilter`。

下列政策提供僅在單一 AWS 帳戶 中在任何目的地資源上建立訂閱篩選條件的許可999999999999：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid":
 "AllowSubscriptionFiltersOnAnyResourceInOneSpecificAccount",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:*"
            ]
        }
    ]
```

```
}
```

下列政策提供僅在單一 AWS 帳戶 sampleDestination中名為 的特定目的地資源上建立訂閱篩選條件的許可123456789012：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Sid": "AllowSubscriptionFiltersOnSpecificResource",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

步驟 4：建立訂閱篩選條件

切換到傳送端帳戶，在此範例中是 111111111111。您現在將在傳送端帳戶中建立訂閱篩選條件。在此範例中，篩選條件與包含 AWS CloudTrail 事件的日誌群組相關聯，因此「根」 AWS 憑證所做的每個記錄活動都會傳送到您先前建立的目的地。如需如何將 AWS CloudTrail 事件傳送至 CloudWatch Logs 的詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的將 CloudTrail 事件傳送至 CloudWatch Logs。

當您輸入以下命令時，確認您以 IAM 使用者身分登入，或是使用您在 步驟 3：新增/驗證跨帳戶目的地的 IAM 許可 中為其新增政策的 IAM 角色登入。

```
aws logs put-subscription-filter \
    --log-group-name "aws-cloudtrail-logs-111111111111-300a971e" \
    --filter-name "firehose_test" \
    --filter-pattern "{$.userIdentity.type = AssumedRole}" \
```

```
    --destination-arn "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
```

日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地可以指向 AWS 資源，例如位於不同區域的 Firehose 串流。

驗證日誌事件的流程

建立訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式的所有傳入日誌事件轉送至 Firehose 交付串流。根據 Firehose 交付串流上設定的時間緩衝間隔，資料開始出現在 Amazon S3 儲存貯體中。一旦經過足夠的時間，您就可以檢查 Amazon S3 儲存貯體來驗證資料。若要檢查儲存貯體，請輸入以下命令：

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

該命令的輸出類似如下：

```
{
    "Contents": [
        {
            "Key": "2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba",
            "LastModified": "2021-02-02T09:00:26+00:00",
            "ETag": "\"EXAMPLEa817fb88fc770b81c8f990d\"",
            "Size": 198,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "firehose+2test",
                "ID": "EXAMPLE27fd05889c665d2636218451970ef79400e3d2aecca3adb1930042e0"
            }
        }
    ]
}
```

然後，您可以輸入下列命令，從儲存貯體擷取特定物件。將 key 的值換成您在前一個命令中找到的值。

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket' --key '2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

在 Amazon S3 物件中的資料會以 gzip 格式壓縮。您可以從命令列使用下列其中一個命令來檢查原始資料：

Linux：

```
zcat testfile.gz
```

macOS：

```
zcat <testfile.gz
```

在執行時間修改目的地成員資格

在某些情況下，您可能需要在您擁有的目的地中新增或移除日誌寄件者。您可以對含新存取政策的目的地使用 PutDestinationPolicy 動作。在下列範例中，之前新增的帳戶 111111111111 會停止傳送更多日誌資料，且帳戶 333333333333 會啟用。

1. 擷取目前與目的地 testDestination 相關聯的政策，並留意 AccessPolicy：

```
aws logs describe-destinations \
    --destination-name-prefix "testFirehoseDestination"

{
    "destinations": [
        {
            "destinationName": "testFirehoseDestination",
            "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
            "roleArn": "arn:aws:iam:: 222222222222:role/CWLtoKinesisFirehoseRole",
            "accessPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Statement
\" : [\n    {\n      \"Sid\" : \"\",\n      \"Effect\" : \"Allow\",\n
 \"Principal\" : {\n        \"AWS\" : \"111111111111 \"\n      },\n      \"Action
\" : \"logs:PutSubscriptionFilter\",\n      \"Resource\" : \"arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination\"\n    }\n  ]\n}\n\n",
            "arn": "arn:aws:logs:us-east-1:
 222222222222:destination:testFirehoseDestination",
            "creationTime": 1612256124430
        }
    ]
}
```

2. 更新政策，以反映帳戶 111111111111 已停用，且帳戶 333333333333 已啟用。將此政策放置在 ~/NewAccessPolicy.json 檔案中：

   JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "333333333333 "
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

3. 使用以下命令將 NewAccessPolicy.json 檔案中定義的政策與目的地建立關聯：

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \

    --access-policy file://~/NewAccessPolicy.json
```

   這最終會停止來自帳戶 ID 111111111111 的日誌事件。在帳戶 333333333333 的擁有者建立訂閱篩選條件後，來自帳戶 ID 333333333333 的日誌事件會立刻開始流向目的地。

使用 Amazon Kinesis Data Streams 的跨帳戶跨區域帳戶層級訂閱

建立跨帳戶訂閱時，您可以指定單一帳戶或一個組織作為寄件者。如果您指定組織，則此程序會讓組織中的所有帳戶都能將日誌傳送至接收者帳戶。

若要跨帳戶共用日誌資料，您需要建立日誌資料寄件者和接收者：

- Log data sender (日誌資料寄件者) - 從收件人取得目的地資訊，並讓 CloudWatch Logs 知道可開始將日誌事件傳送到指定的目的地。在本節其餘部分的程序中，日誌資料寄件者顯示的虛構 AWS 帳戶號碼為 111111111111。

  如果您要讓一個組織中的多個帳戶將日誌傳送至一個收件人帳戶，則可以建立一個政策，授予組織中所有帳戶將日誌傳送至收件人帳戶的許可。您仍然必須為每個寄件者帳戶設定個別的訂閱篩選條件。
- 日誌資料收件人 - 設定封裝 Amazon Kinesis Data Streams 串流的目的地，並讓 CloudWatch Logs 知道收件人想要接收日誌資料。然後，收件人接著會與寄件者共用與其目的地有關的資訊。在本節其餘部分的程序中，日誌資料收件人的虛構 AWS 帳戶號碼為 999999999999。

若要從跨帳戶使用者開始接收日誌事件，日誌資料收件人會先建立 CloudWatch Logs 目的地。每個目的地包含以下關鍵元素：

目的地名稱

您要建立的目的地名稱。

目標 ARN

您要用作訂閱摘要目的地之 AWS 資源的 Amazon Resource Name (ARN)。

角色 ARN

授予 CloudWatch Logs 將資料放入所選串流的必要許可的 AWS Identity and Access Management (IAM) 角色。

存取政策

IAM 政策文件 (JSON 格式，使用 IAM 政策文法撰寫)，決定允許一組使用者寫入您的目的地。

> **Note**
>
> 日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地指向 AWS 的資源可以位於不同的 區域。在以下各節的範例中，區域特定的所有資源都在美國東部 (維吉尼亞北部) 建立。

主題

設定新的跨帳戶訂閱

按照這些章節中的步驟設定新的跨帳戶日誌訂閱。

主題

步驟 1：建立目的地

> **Important**
>
> 此程序中的所有步驟必須在日誌資料收件人帳戶中完成。

在此範例中，日誌資料收件人帳戶的帳戶 AWS ID 為 999999999999，而日誌資料寄件者 AWS 帳戶 ID 為 111111111111。

此範例使用名為 RecipientStream 的 Amazon Kinesis Data Streams 串流建立目的地，以及可讓 CloudWatch Logs 將資料寫入其中的角色。

建立目的地時，CloudWatch Logs 會以收件人帳戶的名義向目的地傳送測試訊息。當訂閱篩選條件在稍後處於作用中狀態時，CloudWatch Logs 會以來源帳戶的名義向目的地傳送日誌事件。

若要建立目的地

1. 在收件人帳戶中，在 Amazon Kinesis Data Streams 中建立目的地串流。在命令提示字元中輸入：

   ```
   aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
   ```

2. 等到 串流變成作用中。您可以使用以下 aws kinesis describe-stream 命令來檢查 StreamDescription.StreamStatus 屬性。此外，應記下 StreamDescription.StreamARN 值，因為稍後要將該值傳遞至 CloudWatch Logs：

   ```
   aws kinesis describe-stream --stream-name "RecipientStream"
   ```

```
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "RecipientStream",
    "StreamARN": "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "340282366920938463463374607431768EXAMPLE",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
 "49551135218688818456679503831981458784591352702 18EXAMPLE"
        }
      }
    ]
  }
}
```

這可能需要花費幾分鐘，讓串流以作用中狀態出現。

3. 建立將授予 CloudWatch Logs 許可以將資料放到串流中的 IAM 角色。首先，您將需要在檔案 ~/TrustPolicyForCWL.json 中建立信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

   此政策包含 aws:SourceArn 全域條件內容金鑰，可指定 sourceAccountId 以協助預防混淆代理人安全問題。如果您在第一次呼叫中還不知道來源帳戶 ID，我們建議您將目的地 ARN 放在來源 ARN 欄位中。在後續呼叫中，應將來源 ARN 設定為從第一次呼叫中收集的實際來源 ARN。如需詳細資訊，請參閱預防混淆代理人。

   ```
   {
       "Statement": {
           "Effect": "Allow",
           "Principal": {
               "Service": "logs.amazonaws.com"
           },
           "Condition": {
               "StringLike": {
                   "aws:SourceArn": [
                       "arn:aws:logs:region:sourceAccountId:*",
                       "arn:aws:logs:region:recipientAccountId:*"
   ```

```
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
}
```

4. 使用 aws iam create-role 命令來建立 IAM 角色，並指定信任政策檔案。記下傳回的 Role.Arn 值，因為稍後也要將此值傳遞至 CloudWatch Logs：

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                },
                "Principal": {
                    "Service": "logs.amazonaws.com"
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2023-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器在檔案 ~/PermissionsForCWL.json 中建立許可政策：

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. 使用 aws iam put-role-policy 命令將許可政策與角色相關聯：

```
aws iam put-role-policy \
    --role-name CWLtoKinesisRole \
    --policy-name Permissions-Policy-For-CWL \
    --policy-document file://~/PermissionsForCWL.json
```

7. 在串流處於作用中狀態且您已建立 IAM 角色後，即可建立 CloudWatch Logs 目的地。

   a. 此步驟不會將存取政策與您的目的地相關聯，且只是完成目的地建立兩步驟中的第一步。請注意在承載中傳回的 DestinationArn：

```
aws logs put-destination \
    --destination-name "testDestination" \
    --target-arn "arn:aws:kinesis:region:999999999999:stream/RecipientStream" \
    --role-arn "arn:aws:iam::999999999999:role/CWLtoKinesisRole"

{
  "DestinationName" : "testDestination",
  "RoleArn" : "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
  "DestinationArn" : "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
  "TargetArn" : "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream"
}
```

   b. 步驟 7a 完成後，即可在日誌資料收件人帳戶中，將存取政策與目的地建立關聯。此政策必須指定 logs:PutSubscriptionFilter 動作，並授予寄件者帳戶許可以存取目的地。

      政策會將許可授予傳送日誌 AWS 的帳戶。您可以在政策中僅指定這一個帳戶，或者如果寄件者帳戶是組織的成員，則政策可以指定組織的組織 ID。如此一來，您可以僅建立一個政策，就能允許一個組織中的多個帳戶將日誌傳送至此目的地帳戶。

使用文字編輯器建立名為 ~/AccessPolicy.json 的檔案，並隨附下列其中一個政策陳述。

此第一個範例政策允許組織中具有 ID 為 o-1234567890 的所有帳戶將日誌傳送至收件人帳戶。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "logs:PutSubscriptionFilter",
                "logs:PutAccountPolicy"
            ],
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-1234567890"
                    ]
                }
            }
        }
    ]
}
```

此下一個範例只允許日誌資料寄件者帳戶 (111111111111) 將日誌傳送至日誌資料收件人帳戶。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111111111111"
            },
            "Action": [
                "logs:PutSubscriptionFilter",
                "logs:PutAccountPolicy"
            ],
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination"
        }
    ]
}
```

c. 將您在上一步驟建立的政策連接到目的地。

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

此存取政策可讓 ID 為 111111111111 的 AWS 帳戶中的使用者針對 ARN arn：aws：logs：*region*：999999999999：destination：testDestination 的目的地呼叫 PutSubscriptionFilter。任何其他使用者針對此目的地呼叫 PutSubscriptionFilter 的嘗試將遭到拒絕。

若要驗證使用者的權限符合存取政策，請參閱《IAM 使用者指南》中的使用政策驗證程式。

完成後，如果您將 AWS Organizations 用於跨帳戶許可，請遵循中的步驟步驟 2：(僅限於使用組織時) 建立 IAM 角色。如果您要將許可直接授予給其他帳戶，而不是使用 Organizations，則可以略過該步驟並繼續進行 步驟 3：建立帳戶層級訂閱篩選條件政策。

步驟 2：(僅限於使用組織時) 建立 IAM 角色

在上一節中，如果您藉由使用授予許可給帳戶 111111111111 所屬組織的存取政策來建立目的地，而不是將許可直接授予給帳戶 111111111111，則按照本節中的步驟進行。若否，則可跳至步驟 步驟 3：建立帳戶層級訂閱篩選條件政策。

本節中的步驟會建立 IAM 角色，CloudWatch 可以代入該角色並驗證寄件者帳戶是否具有針對收件人目的地建立訂閱篩選條件的許可。

在寄件者帳戶中執行此區段中的步驟。角色必須存在於寄件者帳戶中，而且您要在訂閱篩選條件中指定此角色的 ARN。在此範例中，使用者帳戶為 111111111111。

使用 建立跨帳戶日誌訂閱所需的 IAM 角色 AWS Organizations

1. 在檔案 /TrustPolicyForCWLSubscriptionFilter.json 中建立下列信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 建立使用此政策的 IAM 角色。記下命令傳回的 Arn 值，之後在此程序中會用到。在此範例中，我們使用 CWLtoSubscriptionFilterRole 作為要建立的角色的名稱。

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行的動作。

   a. 首先，使用文字編輯器在名為 ~/PermissionsForCWLSubscriptionFilter.json 的檔案中建立下列許可政策。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 輸入下列命令，將您剛建立的許可政策與您在步驟 2 中建立的角色相關聯。

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

完成後，可以繼續進行 步驟 3：建立帳戶層級訂閱篩選條件政策。

步驟 3：建立帳戶層級訂閱篩選條件政策

在建立目的地後，日誌資料收件人帳戶可以與其他 AWS 帳戶共用目的地 ARN (arn:aws:logs:us-east-1:999999999999:destination:testDestination)，讓他們能將日誌事件傳送到相同目的地。然後，這些其他傳送帳戶使用者接著會在個別的日誌群組針對此目的地建立訂閱篩選條件。訂閱篩選條件會立即開始將即時日誌資料從所選的日誌群組傳送到指定的目標。

> **Note**
>
> 如果您要將訂閱篩選條件的許可授予給整個組織，您需要使用您在 步驟 2：(僅限於使用組織時) 建立 IAM 角色 中建立的 IAM 角色 ARN。

在下列範例中，會在傳送帳戶中建立帳戶層級訂閱篩選條件政策。篩選條件與寄件者帳戶相關聯，111111111111因此符合篩選條件和選取條件的每個日誌事件都會傳送至您先前建立的目的地。該目標會封裝名為「RecipientStream」的 串流。

欄位是選用selection-criteria的，但對於排除可能導致訂閱篩選條件無限日誌遞迴的日誌群組非常重要。如需此問題和決定要排除哪些日誌群組的詳細資訊，請參閱 日誌遞迴預防。目前， NOT IN 是 唯一支援的運算子selection-criteria。

```
aws logs put-account-policy \
    --policy-name "CrossAccountStreamsExamplePolicy" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document
 '{"DestinationArn":"arn:aws:logs:region:999999999999:destination:testDestination",
 "FilterPattern": "", "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

寄件者帳戶的日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地可以指向 AWS 資源，例如位於不同區域的 Amazon Kinesis Data Streams 串流。

驗證日誌事件的流動

建立帳戶層級訂閱篩選條件政策後，CloudWatch Logs 會將符合篩選條件模式和選取條件的所有傳入日誌事件，轉送至名為 "RecipientStream" 的目的地串流中封裝的串流。目的地擁有者可以使用 aws kinesis get-shard-iterator 命令來抓取 Amazon Kinesis Data Streams 碎片，並使用 aws kinesis get-records 命令來擷取一些 Amazon Kinesis Data Streams 記錄，以確認是否發生這種情況：

```
aws kinesis get-shard-iterator \
      --stream-name RecipientStream \
      --shard-id shardId-000000000000 \
      --shard-iterator-type TRIM_HORIZON

{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
}

aws kinesis get-records \
      --limit 10 \
      --shard-iterator
      "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

> **Note**
>
> 在 Amazon Kinesis Data Streams 開始傳回資料之前，您可能需要重新執行`get-records`命令幾次。

您應該會看到具有 Amazon Kinesis Data Streams 記錄陣列的回應。Amazon Kinesis Data Streams 記錄中的資料屬性會以 gzip 格式壓縮，然後以 base64 編碼。您可以使用以下 Unix 命令來透過命令列檢查原始資料：

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Base64 解碼和解壓縮資料是以 JSON 形式並以下列結構進行格式化：

```
{
    "owner": "111111111111",
    "logGroup": "CloudTrail/logs",
    "logStream": "111111111111_CloudTrail/logs_us-east-1",
    "subscriptionFilters": [
        "RecipientStream"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        }
    ]
}
```

資料結構中的關鍵元素如下：

messageType

資料訊息將使用「DATA_MESSAGE」類型。有時 CloudWatch Logs 可能會發出 "CONTROL_MESSAGE" 類型的 Amazon Kinesis Data Streams 記錄，主要用於檢查目的地是否可連線。

owner

原始日誌資料 AWS 的帳戶 ID。

logGroup

原始日誌資料的日誌群組名稱。

logStream

原始日誌資料的日誌串流名稱。

subscriptionFilters

與原始日誌資料相符的訂閱篩選條件名稱清單。

logEvents

實際的日誌資料，以一系列的日誌事件記錄呈現。「id」屬性是每個記錄事件的唯一識別符。

policyLevel

強制執行政策的層級。"ACCOUNT_LEVEL_POLICY" 是帳戶層級訂閱篩選條件政策`policyLevel`的 。

在執行期修改目的地成員資格

您可能遇到以下情況：您需要從擁有的目的地中新增或移除某些使用者的成員資格。您可以在目的地使用 `put-destination-policy` 命令並指定新的存取政策。在下列範例中，之前新增帳戶 111111111111 會停止傳送任何更多資料，且帳戶 222222222222 會啟用。

1. 擷取目前與目的地 testDestination 相關聯的政策，並留意 AccessPolicy：

```
aws logs describe-destinations \
    --destination-name-prefix "testDestination"

{
 "Destinations": [
   {
     "DestinationName": "testDestination",
     "RoleArn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
     "DestinationArn":
 "arn:aws:logs:region:999999999999:destination:testDestination",
     "TargetArn": "arn:aws:kinesis:region:999999999999:stream/RecipientStream",
```

```
    "AccessPolicy": "{\"Version\": \"2012-10-17\", \"Statement\":
 [{\"Sid\": \"\", \"Effect\": \"Allow\", \"Principal\": {\"AWS\":
 \"111111111111\"}, \"Action\": \"logs:PutSubscriptionFilter\", \"Resource\":
 \"arn:aws:logs:region:999999999999:destination:testDestination\"}] }"
   }
 ]
}
```

2. 更新政策，以反映帳戶 111111111111 已停用，且帳戶 222222222222 已啟用。將此政策放置在 ~/NewAccessPolicy.json 檔案中：

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "222222222222"
            },
            "Action": [
                "logs:PutSubscriptionFilter",
                "logs:PutAccountPolicy"
            ],
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination"
        }
    ]
}
```

3. 呼叫 PutDestinationPolicy 以與含目的地之 NewAccessPolicy.json 檔案中定義的政策相關聯：

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

   這最終會從帳戶 ID 111111111111 停用日誌事件。在帳戶 222222222222 的擁有者建立訂閱篩選條件後，來自帳戶 ID 222222222222 的日誌事件會立刻開始流向目的地。

更新現有的跨帳戶訂閱

如果您目前有跨帳戶日誌訂閱，其中目的地帳戶僅授予特定寄件者帳戶許可，而您想要更新此訂閱，讓目的地帳戶授予組織中所有帳戶的存取權，請按照本節中的步驟進行。

主題

步驟 1：更新訂閱篩選條件

> **Note**
>
> 只有跨帳戶訂閱由 從 AWS 服務啟用記錄 所列服務建立的日誌才需要此步驟。如果您不使用這些日誌群組之一建立的日誌，則可以跳至 步驟 2：更新現有的目的地存取政策。

在某些情況下，您必須更新所有傳送日誌至目的地帳戶的寄件者帳戶中的訂閱篩選條件。此更新會新增 IAM 角色，CloudWatch 可以代入該角色並驗證寄件者帳戶是否具有將日誌傳送至收件人帳戶的許可。

針對您想要更新的每個寄件者帳戶，按照本節中的步驟進行，以將組織 ID 用於跨帳戶訂閱許可。

在本節的範例中，111111111111 和 222222222222 兩個帳戶已經建立訂閱篩選條件，以將日誌傳送至帳戶 999999999999。現有的訂閱篩選條件值如下：

```
## Existing Subscription Filter parameter values
{
    "DestinationArn": "arn:aws:logs:region:999999999999:destination:testDestination",
    "FilterPattern": "{$.userIdentity.type = Root}",
    "Distribution": "Random"
}
```

如果需要尋找目前的訂閱篩選條件參數值，請輸入下列命令。

```
aws logs describe-account-policies \
--policy-type "SUBSCRIPTION_FILTER_POLICY" \
--policy-name "CrossAccountStreamsExamplePolicy"
```

更新訂閱篩選條件以開始將組織 ID 用於跨帳戶日誌許可

1. 在檔案 ~/TrustPolicyForCWL.json 中建立下列信任政策。使用文字編輯器來建立此政策檔案，請勿使用 IAM 主控台。

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 建立使用此政策的 IAM 角色。記下命令傳回的 Arn 值的 Arn 值，之後在此程序中會用到。在此範例中，我們使用 CWLtoSubscriptionFilterRole 作為要建立的角色的名稱。

```
aws iam create-role
    \ --role-name CWLtoSubscriptionFilterRole
    \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行的動作。

   a. 首先，使用文字編輯器在名為 /PermissionsForCWLSubscriptionFilter.json 的檔案中建立下列許可政策。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 輸入下列命令，將您剛建立的許可政策與您在步驟 2 中建立的角色相關聯。

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. 輸入下列命令以更新訂閱篩選條件政策。

```
aws logs put-account-policy \
    --policy-name "CrossAccountStreamsExamplePolicy" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document
 '{"DestinationArn":"arn:aws:logs:region:999999999999:destination:testDestination",
 "FilterPattern": "{$.userIdentity.type = Root}", "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

步驟 2：更新現有的目的地存取政策

更新所有寄件者帳戶中的訂閱篩選條件之後，您可以更新收件人帳戶中的目的地存取政策。

下列範例中，收件人帳戶為 999999999999，且目的地名稱為 testDestination。

此更新可讓屬於組織且 ID 為 o-1234567890 的所有帳戶傳送日誌至收件人帳戶。只有已建立訂閱篩選條件的帳戶才會真的傳送日誌至收件人帳戶。

更新收件人帳戶中的目的地存取政策，以開始將組織 ID 用於許可

1. 在收件人帳戶中，使用文字編輯器建立 ~/AccessPolicy.json 檔案，其中包含以下內容。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "logs:PutSubscriptionFilter",
                "logs:PutAccountPolicy"
            ],
            "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-1234567890"
                    ]
                }
            }
        }
    ]
}
```

2. 輸入下列命令，將您剛建立的政策連接到現有的目的地。若要更新目的地以使用具有組織 ID 的存取政策，而非列出特定 AWS 帳戶 IDs的存取政策，請包含 `force` 參數。

> **Warning**
>
> 如果您使用 中列出的 AWS 服務傳送的日誌從 AWS 服務啟用記錄，則在執行此步驟之前，您必須先更新所有寄件者帳戶中的訂閱篩選條件，如 中所述步驟 1：更新訂閱篩選條件。

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
    \ --force
```

使用 Firehose 的跨帳戶跨區域帳戶層級訂閱

若要跨帳戶共用日誌資料，您需要建立日誌資料寄件者和接收者：

- 日誌資料寄件者 - 從收件人取得目的地資訊，並讓 CloudWatch Logs 知道可開始將日誌事件傳送到指定的目的地。在本節其餘部分的程序中，日誌資料寄件者顯示的虛構 AWS 帳戶號碼為 111111111111。
- 日誌資料收件人 - 設定封裝 Amazon Kinesis Data Streams 串流的目的地，並讓 CloudWatch Logs 知道收件人想要接收日誌資料。然後，收件人接著會與寄件者共用與其目的地有關的資訊。在本節其餘部分的程序中，日誌資料收件人的虛構 AWS 帳戶號碼為 222222222222。

本節中的範例使用 Firehose 交付串流搭配 Amazon S3 儲存。您也可以使用不同的設定來設定 Firehose 交付串流。如需詳細資訊，請參閱建立 Firehose 交付串流。

> **Note**
>
> 日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地指向 AWS 的資源可以位於不同的 區域。

> **Note**
>
> 支援相同帳戶和跨區域交付串流的 Firehose 訂閱篩選條件。

主題

步驟 1：建立 Firehose 交付串流

> **Important**
>
> 在完成下列步驟之前，您必須使用存取政策，以便 Firehose 可以存取您的 Amazon S3 儲存貯體。如需詳細資訊，請參閱《Amazon Data Firehose 開發人員指南》中的控制存取。
> 必須在日誌資料收件人帳戶中完成本區段 (步驟 1) 中的所有步驟。
> 在以下範例命令中使用美國東部 (維吉尼亞北部)。請將此區域替換成您部署的正確區域。

建立要用作目的地的 Firehose 交付串流

1. 建立 Amazon S3 儲存貯體：

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-1
```

2. 建立 IAM 角色，授予 Firehose 將資料放入儲存貯體的許可。

a. 首先，使用文字編輯器在檔案 ~/TrustPolicyForFirehose.json 中建立信任政策。

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
 "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
 { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
```

b. 建立 IAM 角色，並指定您剛建立的信任政策檔案。

```
aws iam create-role \
    --role-name FirehosetoS3Role \
    --assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

c. 此命令的輸出看起來如下：記下角色名稱和角色 ARN。

```
{
    "Role": {
        "Path": "/",
        "RoleName": "FirehosetoS3Role",
        "RoleId": "AROAR3BXASEKW7K635M53",
        "Arn": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
        "CreateDate": "2021-02-02T07:53:10+00:00",
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Effect": "Allow",
                "Principal": {
                    "Service": "firehose.amazonaws.com"
                },
                "Action": "sts:AssumeRole",
                "Condition": {
                    "StringEquals": {
                        "sts:ExternalId": "222222222222"
                    }
                }
            }
        }
    }
}
```

3. 建立許可政策，以定義 Firehose 可以在您的帳戶中執行的動作。

a. 首先，使用文字編輯器在名為 ~/PermissionsForFirehose.json 的檔案中建立下列許可政策。根據您的使用案例，可能需要為此檔案新增更多許可。

```
{
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket",
            "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
    }]
}
```

b. 輸入下列命令，將您剛建立的許可政策與 IAM 角色相關聯。

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. 輸入下列命令來建立 Firehose 交付串流。以部署的正確值取代 *my-role-arn* 和 *amzn-s3-demo-bucket2-arn*。

```
aws firehose create-delivery-stream \
   --delivery-stream-name 'my-delivery-stream' \
   --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role", "BucketARN":
 "arn:aws:s3:::amzn-s3-demo-bucket"}'
```

輸出格式應類似以下內容：

```
{
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

步驟 2：建立目的地

> **Important**
>
> 此程序中的所有步驟必須在日誌資料收件人帳戶中完成。

建立目的地時，CloudWatch Logs 會以收件人帳戶的名義向目的地傳送測試訊息。當訂閱篩選條件在稍後處於作用中狀態時，CloudWatch Logs 會以來源帳戶的名義向目的地傳送日誌事件。

若要建立目的地

1. 等到您在 中建立的 Firehose 串流步驟 1：建立 Firehose 交付串流變成作用中。您可以使用以下命令來檢查 StreamDescription.StreamStatus 屬性。

   ```
   aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
   ```

   此外，請留意 DeliveryStreamDescription.DeliveryStreamARN 值，因後續步驟中需要用到。此命令的範例輸出：

   ```
   {
       "DeliveryStreamDescription": {
           "DeliveryStreamName": "my-delivery-stream",
           "DeliveryStreamARN": "arn:aws:firehose:us-
   east-1:222222222222:deliverystream/my-delivery-stream",
           "DeliveryStreamStatus": "ACTIVE",
           "DeliveryStreamEncryptionConfiguration": {
               "Status": "DISABLED"
           },
           "DeliveryStreamType": "DirectPut",
           "VersionId": "1",
           "CreateTimestamp": "2021-02-01T23:59:15.567000-08:00",
           "Destinations": [
               {
                   "DestinationId": "destinationId-000000000001",
                   "S3DestinationDescription": {
                       "RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
                       "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
                       "BufferingHints": {
                           "SizeInMBs": 5,
                           "IntervalInSeconds": 300
   ```

```
                    },
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    }
                },
                "ExtendedS3DestinationDescription": {
                    "RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
                    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
                    "BufferingHints": {
                        "SizeInMBs": 5,
                        "IntervalInSeconds": 300
                    },
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    },
                    "S3BackupMode": "Disabled"
                }
            }
        ],
        "HasMoreDestinations": false
    }
}
```

可能需要花費幾分鐘，交付串流才會變成作用中狀態。

2. 當交付串流處於作用中狀態時，請建立 IAM 角色，以授予 CloudWatch Logs 將資料放入 Firehose 串流的許可。首先，您將需要在檔案 ~/TrustPolicyForCWL.json 中建立信任政策。請使用文字編輯器來建立此政策。如需 CloudWatch Logs 端點的詳細資訊，請參閱 Amazon CloudWatch Logs 端點和配額。

   此政策包含 aws:SourceArn 全域條件內容金鑰，可指定 sourceAccountId 以協助預防混淆代理人安全問題。如果您在第一次呼叫中還不知道來源帳戶 ID，我們建議您將目的地 ARN 放在來源 ARN 欄位中。在後續呼叫中，應將來源 ARN 設定為從第一次呼叫中收集的實際來源 ARN。如需詳細資訊，請參閱預防混淆代理人。

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
     }
}
```

3. 使用 aws iam create-role 命令來建立 IAM 角色，並指定您剛建立的信任政策檔案。

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

下列為範例輸出。請留意傳回的 Role.Arn 值，因為後續步驟中需要用到。

```
{
    "Role": {
        "Path": "/",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "RoleId": "AROAR3BXASEKYJYWF243H",
        "Arn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
        "CreateDate": "2023-02-02T08:10:43+00:00",
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Action": "sts:AssumeRole",
                "Condition": {
                    "StringLike": {
```

```
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. 建立許可政策以定義 CloudWatch Logs 可在您的帳戶上執行哪些動作。首先，使用文字編輯器在檔案 ~/PermissionsForCWL.json 中建立許可政策：

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
}
```

5. 輸入以下命令，將許可政策與角色建立關聯：

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Firehose 交付串流處於作用中狀態且您已建立 IAM 角色後，您可以建立 CloudWatch Logs 目的地。

   a. 此步驟不會將存取政策與您的目的地建立關聯，且為完成建立目的地之兩個步驟的僅第一個步驟。記下承載中傳回的新目的地的 ARN，因為您會在後續步驟中使用它作為 `destination.arn`。

```
aws logs put-destination \

    --destination-name "testFirehoseDestination" \
    --target-arn "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-
delivery-stream" \
    --role-arn "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole"
```

```
{
    "destination": {
        "destinationName": "testFirehoseDestination",
        "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
        "roleArn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
        "arn": "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"}
}
```

b. 上一個步驟完成後，請在日誌資料收件人帳戶中 (222222222222)，將存取政策與目的地建立關聯。此政策可讓日誌資料寄件者帳戶 (111111111111) 只能在日誌資料收件人帳戶 (222222222222) 中存取目的地。您可以使用文字編輯器，將此政策放入 ~/AccessPolicy.json檔案中：

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

c. 這會建立可定義誰擁有對目的地的寫入存取權之政策。此政策必須指定logs:PutSubscriptionFilter和 logs:PutAccountPolicy動作才能存取目的地。跨帳戶使用者將使用 PutSubscriptionFilter和 PutAccountPolicy動作，將日誌事件傳送至目的地。

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
```

```
    --access-policy file://~/AccessPolicy.json
```

步驟 3：建立帳戶層級訂閱篩選條件政策

切換到傳送端帳戶，在此範例中是 111111111111。您現在將在傳送帳戶中建立帳戶層級訂閱篩選條件政策。在此範例中，篩選條件會導致除了兩個日誌群組之外ERROR，所有包含字串的每個日誌事件，都交付到您先前建立的目的地。

```
aws logs put-account-policy \
    --policy-name "CrossAccountFirehoseExamplePolicy" \
    --policy-type "SUBSCRIPTION_FILTER_POLICY" \
    --policy-document '{"DestinationArn":"arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination", "FilterPattern":
 "{$.userIdentity.type = AssumedRole}", "Distribution": "Random"}' \
    --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
 "LogGroupToExclude2"]' \
    --scope "ALL"
```

傳送帳戶的日誌群組和目的地必須位於相同的 AWS 區域。不過，目的地可以指向 AWS 資源，例如位於不同區域的 Firehose 串流。

驗證日誌事件的流程

建立訂閱篩選條件後，CloudWatch Logs 會將符合篩選條件模式和選取條件的所有傳入日誌事件轉送至 Firehose 交付串流。根據 Firehose 交付串流上設定的時間緩衝間隔，資料開始出現在 Amazon S3 儲存貯體中。一旦經過足夠的時間，您就可以檢查 Amazon S3 儲存貯體來驗證資料。若要檢查儲存貯體，請輸入以下命令：

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

該命令的輸出類似如下：

```
{
    "Contents": [
        {
            "Key": "2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba",
            "LastModified": "2023-02-02T09:00:26+00:00",
            "ETag": "\"EXAMPLEa817fb88fc770b81c8f990d\"",
            "Size": 198,
```

```
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "firehose+2test",
                "ID": "EXAMPLE27fd05889c665d2636218451970ef79400e3d2aecca3adb1930042e0"
            }
        }
    ]
}
```

然後，您可以輸入下列命令，從儲存貯體擷取特定物件。將 key 的值換成您在前一個命令中找到的值。

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket' --key '2021/02/02/08/my-delivery-stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

在 Amazon S3 物件中的資料會以 gzip 格式壓縮。您可以從命令列使用下列其中一個命令來檢查原始資料：

Linux：

```
zcat testfile.gz
```

macOS：

```
zcat <testfile.gz
```

在執行時間修改目的地成員資格

在某些情況下，您可能需要在您擁有的目的地中新增或移除日誌寄件者。您可以使用 PutDestinationPolicy 和目的地上的PutAccountPolicy動作搭配新的存取政策。在下列範例中，之前新增的帳戶 111111111111 會停止傳送更多日誌資料，且帳戶 333333333333 會啟用。

1. 擷取目前與目的地 testDestination 相關聯的政策，並留意 AccessPolicy：

   ```
   aws logs describe-destinations \
       --destination-name-prefix "testFirehoseDestination"
   ```

   傳回的資料可能如下所示。

   ```
   {
   ```

```
    "destinations": [
        {
            "destinationName": "testFirehoseDestination",
            "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
            "roleArn": "arn:aws:iam:: 222222222222:role/CWLtoKinesisFirehoseRole",
            "accessPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Statement
\" : [\n    {\n      \"Sid\" : \"\",\n      \"Effect\" : \"Allow\",\n
 \"Principal\" : {\n        \"AWS\" : \"111111111111 \"\n      },\n      \"Action
\" : \"logs:PutSubscriptionFilter\",\n      \"Resource\" : \"arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination\"\n    }\n  ]\n}\n\n",
            "arn": "arn:aws:logs:us-east-1:
 222222222222:destination:testFirehoseDestination",
            "creationTime": 1612256124430
        }
    ]
}
```

2. 更新政策，以反映帳戶 111111111111 已停用，且帳戶 333333333333 已啟用。將此政策放置在 ~/NewAccessPolicy.json 檔案中：

   JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "333333333333 "
      },
      "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

3. 使用以下命令將 NewAccessPolicy.json 檔案中定義的政策與目的地建立關聯：

```
aws logs put-destination-policy \
```

```
    --destination-name "testFirehoseDestination" \

    --access-policy file://~/NewAccessPolicy.json
```

這最終會停止來自帳戶 ID 111111111111 的日誌事件。在帳戶 333333333333 的擁有者建立訂閱篩選條件後，來自帳戶 ID 333333333333 的日誌事件會立刻開始流向目的地。

預防混淆代理人

混淆代理人問題屬於安全性議題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在 中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了防止這種情況， AWS 提供工具，協助您保護所有 服務的資料，讓服務主體能夠存取您帳戶中的資源。

我們建議在資源政策中使用 aws:SourceArn、aws:SourceOrgID、 aws:SourceAccount和 aws:SourceOrgPaths全域條件內容索引鍵，以限制將另一個服務提供給資源的許可。使用 aws:SourceArn，僅將一個資源與跨服務存取權相關聯。使用 aws:SourceAccount，讓該帳戶中的任何資源都與跨服務使用相關聯。使用 aws:SourceOrgID，允許組織內任何帳戶的任何資源與跨服務使用相關聯。使用 aws:SourceOrgPaths，將 AWS Organizations 路徑中帳戶的任何資源與跨服務使用相關聯。如需使用和了解路徑的詳細資訊，請參閱了解 AWS Organizations 實體路徑。

防範混淆代理人問題的最有效方法是使用 aws:SourceArn 全域條件內容索引鍵，以及資源的完整 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 aws:SourceArn 全域內容條件索引鍵搭配萬用字元 (*) 來表示 ARN 的未知部分。例如 arn:aws:*servicename*:*:*123456789012*:*。

如果 aws:SourceArn 值不包含帳戶 ID (例如 Amazon S3 儲存貯體 ARN)，您必須同時使用 aws:SourceAccount 和 aws:SourceArn 來限制許可。

若要大規模防範混淆代理人問題，請在資源型政策中使用 aws:SourceOrgID 或 aws:SourceOrgPaths 全域條件內容鍵和資源的組織 ID 或組織路徑。當您新增、移除或移動組織中的帳戶時，包含 aws:SourceOrgID 或 aws:SourceOrgPaths 鍵的政策將會自動包含正確的帳戶，您無需手動更新政策。

記載的政策用於授予對 CloudWatch Logs 的存取權，以將資料寫入 中的 Amazon Kinesis Data Streams 和 Firehose，步驟 1：建立目的地並步驟 2：建立目的地示範如何使用aws:SourceArn全域條件內容金鑰來協助防止混淆代理人問題。

日誌遞迴預防

如果未防止，訂閱篩選條件可能會導致 CloudWatch Logs 和目的地的擷取計費大幅增加，則存在導致無限日誌遞迴的風險。當訂閱篩選條件與因訂閱交付工作流程而接收日誌事件的日誌群組相關聯時，就會發生這種情況。擷取到日誌群組的日誌會傳送到目的地，導致日誌群組擷取更多日誌，然後再次轉送到目的地，建立遞迴迴圈。

例如，將目的地為 Firehose 的訂閱篩選條件視為 Firehose，這會將日誌事件交付給 Amazon S3。此外，也有 Lambda 函數可處理交付至 Amazon S3 的新事件，並產生一些日誌本身。如果訂閱篩選條件套用至 Lambda 函數的日誌群組，則函數產生的日誌事件會在目的地轉送至 Firehose 和 Amazon S3，然後再次叫用該函數，導致產生更多日誌並轉送至 Firehose 和 Amazon S3，進而再次叫用該函數等。這將發生在無限迴圈中，導致日誌擷取、Firehose 和 Amazon S3 的計費意外增加。

如果 Lambda 函數連接到已啟用 CloudWatch Logs 流程日誌的 VPC，則 VPC 的日誌群組也可能導致日誌遞迴。

建議您不要將訂閱篩選條件套用至屬於訂閱交付工作流程一部分的日誌群組。對於帳戶層級訂閱篩選條件，請使用 `PutAccountPolicy` API 中的 `selectionCriteria` 參數，從政策中排除這些日誌群組。

排除日誌群組時，請考慮下列產生日誌的服務，這些 AWS 服務可能是訂閱交付工作流程的一部分：

- Amazon EC2 搭配 Fargate
- Lambda
- AWS 步驟函數
- 為 CloudWatch Logs 啟用的 Amazon VPC 流程日誌

> **Note**
>
> Lambda 目的地的日誌群組所產生的日誌事件將不會轉送回帳戶層級訂閱篩選條件政策的 Lambda 函數。在此情況下，帳戶訂閱政策`selectionCriteria`不需要使用 排除目的地 Lambda 函數的日誌群組。

用於指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法

> **Note**
>
> 如需有關如何使用 Amazon CloudWatch Logs Insights 查詢語言查詢日誌群組的資訊，請參閱 CloudWatch Logs Insights 語言查詢語法。

使用 CloudWatch Logs，您可以使用指標篩選條件將日誌資料轉換為可操作的指標、將日誌事件路由至 AWS 其他服務的訂閱篩選條件、篩選日誌事件以搜尋日誌事件，以及 Live Tail 在擷取日誌時以互動方式即時檢視日誌。

篩選條件模式構成指標篩選條件、訂閱篩選條件、日誌事件和 Live Tail 用來比對日誌事件中詞彙的語法。詞彙可以是單字、完全相符片語或數值。規則運算式 (regex) 可用於建立獨立的篩選條件模式，或與 JSON 和以空格分隔的篩選條件模式整合。

使用要比對的詞彙建立篩選條件模式。篩選條件模式僅傳回包含您所定義的詞彙的日誌事件。您可以在 CloudWatch 主控台測試篩選條件模式。

主題

支援的規則運算式 (regex) 語法

支援的 regex 語法

使用 regex 搜尋和篩選日誌資料時，必須用 % 括住運算式。

包含 regex 的篩選條件模式只能包括下列字元：

- 英數字元 – 英數字元包含字母 (A 到 Z 或 a 到 z) 或數字 (0 到 9) 字元。

- 支援的符號字元 – 這些字元包括：':'、'_'、'#'、'='、'@'、'/'、';'、',' 和 '-'。例如，由於不支援 '!'，因此 %something!% 會遭拒。
- 支援的運算子 - 包括：'^'、'$'、'?'、'['、']'、'{'、'}'、'|'、'\'、'*'、'+' 和 '.'。

不支援運算子（ 和 ）。您不可以使用括號來定義子模式。

不支援多位元組字元。

> **Note**
>
> 配額
> 建立指標篩選條件或訂閱篩選條件時，每個日誌群組最多有 5 個包含 regex 的篩選條件模式。在為指標篩選條件和訂閱篩選條件建立分隔或 JSON 篩選條件模式，或在篩選條件日誌事件或 Live Tail 時，每個篩選模式限制為 2 個 regex。

受支援運算子的使用

- ^：將比對錨定到字串的開頭。例如，%^[hc]at% 比對 "hat" 和 "cat"，但只比對字串的開頭。
- $：將比對錨定到字串的結尾。例如，%[hc]at$% 比對 "hat" 和 "cat"，但只比對字串的結尾。
- ?：符合前一個字詞的零或一次出現。例如，%colou?r% 可同時比對 "color" 和 "colour"。
- []：定義字元類別。比對包含在括號內的字元清單或字元範圍。例如，%[abc]% 比對 "a"、"b" 或 "c"；%[a-z]% 比對從 "a" 到 "z" 的任何小寫字元；以及 %[abcx-z]% 比對 "a"、"b"、"c"、"x"、"y" 或 "z"。
- {m, n}：比對前一個詞彙至少 m 次且不超過 n 次。例如，%a{3,5}% 僅比對 "aaa"、"aaaa" 和 "aaaaa"。

  > **Note**
  >
  > 如果您選擇不定義下限或上限，則可省略 m 或 n。

- |：布林值 "Or"，比對垂直列任一側的詞彙。例如：
  - %gra|ey% 可以比對「灰色」或「灰色」
  - %^starting|^initializing|^shutting down% 可以比對「開始 ...」或「初始化 ...」或「關閉」，但不符合「略過初始化 ...」
  - %abcc|ab[^c]$ 可以比對「abcc ...」和「aba ...」，但不符合「aac ...」

- `\`：逸出字元，可讓您使用運算子的文字含義，而非其特殊含義。例如，因為括號被逸出，`%\[.\]%` 比對任何用 "[" 和 "]" 括住的單一字元，例如 "[a]"、"[b]"、"[7]"、"[@]"、"[]]" 和 "[ ]"。

  > **Note**
  >
  > `%10\.10\.0\.1%` 是建立可比對 IP 地址 10.10.0.1 的 regex 的正確方法。

- `*`：符合先前期限的零個或多個執行個體。例如， `%ab*c%` 可比對 "ac"、"abc" 和 "abbbc"；`%ab[0-9]*%` 可比對 "ab"、"ab0" 和 "ab129"。
- `+`：符合先前期限的一或多個執行個體。例如，`%ab+c%` 可比對 "abc"、"abbc" 和 "abbbc"，而非 "ac"。
- `.`：比對任一個單一字元。例如，`%.at%` 比對以 "at" 結尾的任意三個字串，包括 "hat"、"cat"、"bat"、"4at"、"#at" 和 "at" (以空格開始)。

  > **Note**
  >
  > 建立比對 IP 地址的 regex 時，逸出 `.` 運算子很重要。例如，`%10.10.0.1%` 可比對 "10010,051"，這可能不是運算式的實際預期用途。

- `\d`、`\D`：比對數字/非數字字元。例如，`%\d%` 等於 `%[0-9]%`，`%\D%` 等於 `%[^0-9]%`。

  > **Note**
  >
  > 大寫運算子表示其對應小寫運算子的反轉。

- `\s`、`\S`：比對空白字元/非空白字元。

  > **Note**
  >
  > 大寫運算子表示其對應小寫運算子的反轉。空白字元包括 tab (`\t`)、空格 ( ) 和換行符 (`\n`) 字元。

- `\w`、`\W`：比對英數字元/非英數字元。例如，`%\w%` 等於 `%[a-zA-Z_0-9]%`，`%\W%` 等於 `%[^a-zA-Z_0-9]%`。

  > **Note**
  >
  > 大寫運算子表示其對應小寫運算子的反轉。

- \xhh：比對兩位數十六進位字元的 ASCII 映射。\x 為逸出序列，表示下列字元代表 ASCII 的十六進位值。hh 指定指向 ASCII 資料表中某個字元的兩個十六進位數字 (0-9 和 A-F)。

> **Note**
>
> 您可以使用 \xhh 來比對篩選條件模式不支援的符號字元。例如，%\x3A% 比對：；%\x28% 比對 (。

使用篩選條件模式來比對詞彙與規則運算式 (regex)

使用 regex 比對詞彙

您可以使用以 % (regex 模式之前和之後的百分比符號) 括住的 regex 模式來比對日誌事件中的術語。以下程式碼片段為篩選條件模式的範例，其會傳回包含 AUTHORIZED 關鍵字的所有日誌事件。

如需支援的規則運算式清單，請參閱支援的規則運算式。

```
%AUTHORIZED%
```

此篩選條件模式會傳回如下日誌事件訊息：

- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[SUCCESS 200] AUTHORIZED REQUEST`

使用篩選條件模式來比對日誌事件中的詞彙

在非結構化日誌事件中比對詞彙

下列程式碼片段範例示範了如何使用篩選條件模式，來比對非結構化日誌事件中的詞彙。

> **Note**
>
> 篩選條件模式區分大小寫。將完全相符字詞和包含非英數字元的詞彙括在雙引號 ("") 中。

Example: Match a single term

下方程式碼片段為單一詞彙篩選條件模式的範例，其會傳回所有訊息中包含單字 ERROR (錯誤) 的日誌事件。

```
ERROR
```

此篩選條件模式會比對以下日誌事件訊息：

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

Example: Match multiple terms

以下程式碼片段為多項詞彙篩選條件範例，其會傳回所有訊息中包含單字 ERROR (錯誤) 和 ARGUMENTS (引數) 的日誌事件。

```
ERROR ARGUMENTS
```

篩選條件會傳回如下的日誌事件訊息：

- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

此篩選條件模式不會傳回以下日誌事件訊息，因為其不包含在篩選條件模式中指定的兩個詞彙。

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`

Example: Match optional terms

您可以使用模式比對，來建立會傳回包含選用詞彙之日誌事件的篩選條件模式。將問號 ("?") 置於想要比對的詞彙之前。下列程式碼片段為篩選條件模式的範例，篩選條件模式會傳回所有訊息中包含單字 ERROR 或單字 ARGUMENTS 的日誌事件。

```
?ERROR ?ARGUMENTS
```

此篩選條件模式會比對以下日誌事件訊息：

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

> **Note**
>
> 您無法將問號 ("？") 與其他篩選條件模式結合，例如包含和排除詞彙。如果您將 "？" 與其他篩選條件模式結合，則會忽略所有問號詞彙。
> 例如，下列篩選條件模式符合包含單字 的所有事件`REQUEST`，但問號 ("？") 篩選條件詞彙會被忽略且沒有效果。
>
> ```
> ?ERROR ?ARGUMENTS REQUEST
> ```
>
> 日誌事件相符
>
> - `[INFO] REQUEST FAILED`
> - `[WARN] UNAUTHORIZED REQUEST`
> - `[ERROR] 400 BAD REQUEST`

Example: Match exact phrases

下列程式碼片段為篩選條件的範例，其會傳回訊息中包含完全相符字詞 INTERNAL SERVER ERROR (內部伺服器錯誤) 的日誌事件。

```
"INTERNAL SERVER ERROR"
```

此篩選條件模式會傳回以下日誌事件訊息：

- `[ERROR 500] INTERNAL SERVER ERROR`

Example: Include and exclude terms

您可以建立篩選條件模式，令其傳回訊息中包含某些詞彙，並排除其他詞彙的日誌事件。將減號 ("-") 置於想要排除的詞彙之前。下列程式碼片段為篩選條件模式的範例，其會傳回訊息中包含詞彙 ERROR (錯誤) 並排除詞彙 ARGUMENTS (引數) 的日誌事件。

```
ERROR -ARGUMENTS
```

此篩選條件模式會傳回如下日誌事件訊息：

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`

此篩選條件模式不會傳回以下日誌事件訊息，因為其包含單字 ARGUMENTS。

- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

Example: Match everything

您可以用雙引號比對日誌事件中的所有內容。下列程式碼片段為篩選條件模式的範例，其會傳回所有日誌事件。

```
" "
```

使用篩選條件模式來比對 JSON 日誌事件中的詞彙

編寫 JSON 日誌事件的篩選條件模式

下列範例示範了如何寫入篩選條件模式的語法，來比對包含字串和數值和 JSON 詞彙。

Writing filter patterns that match strings

您可以建立篩選條件模式以比對 JSON 日誌事件中的字串。下列程式碼片段顯示的範例為以字串為基礎的篩選條件模式。

```
{ PropertySelector EqualityOperator String }
```

用大括號 ("{}") 括住篩選條件模式。以字串為基礎的篩選條件模式必須包含以下部分：

- Property selector (屬性選擇器)

  屬性選擇器以後面帶一個句點的貨幣符號 ("$.") 開始。屬性選擇器都是英數字元字串，並支援連字號 ("-") 和底線 ("_") 字元。字串不支援科學符號。屬性選擇器指向 JSON 日誌事件中的值節點。值節點可以是字串或數字。將陣列放在屬性選擇器之後。陣列中的元素依循從零開始的編號系統，意即陣列中的第一個元素是元素 0，第二個元素是元素 1，依此類推。將元素括在中括號內 ("[]")。如果屬性選擇器指向陣列或物件，則篩選條件模式會與日誌格式不相符。如果 JSON 屬性包含句點 ("."")，則可以使用括號標記法來選取該屬性。

  > **Note**
  >
  > 萬用字元選取器
  > 您可以使用 JSON 萬用字元來選取任何陣列元素或任何 JSON 物件欄位。
  > 配額
  > 在屬性選取器中，您只能使用最多一個萬用字元選取器。

- Equality operator (等式運算子)

等式運算子以下列符號之一開始：等於 ("=") 或不等於 ("!=")。等式運算子傳回布林值 (true 或 false)。

- 字串

  您可以用雙引號 ("") 括住字串。包含除了英數字元和底線符號以外類型的字串必須置於雙引號中。使用星號 ("*") 作為萬用字元來比對文本。

  > **Note**
  >
  > 建立篩選條件模式來比對 JSON 日誌事件中的詞彙時，您可以使用任何條件式規則運算式。如需支援的規則運算式清單，請參閱支援的規則運算式。

下列程式碼片段包含一個篩選條件模式範例，示範如何設定篩選條件模式的格式，使用字串來比對 JSON 詞彙。

```
{ $.eventType = "UpdateTrail" }
```

Writing filter patterns that match numeric values

您可以建立篩選條件模式以比對 JSON 日誌事件中的數值。下列程式碼片段為與數值相符之篩選條件模式的語法範例。

```
{ PropertySelector NumericOperator Number }
```

用大括號 ("{}") 括住篩選條件模式。比對數值的篩選條件模式必須包含以下部分：

- Property selector (屬性選擇器)

  屬性選擇器以後面帶一個句點的貨幣符號 ("$.") 開始。屬性選擇器都是英數字元字串，並支援連字號 ("-") 和底線 ("_") 字元。字串不支援科學符號。屬性選擇器指向 JSON 日誌事件中的值節點。值節點可以是字串或數字。將陣列放在屬性選擇器之後。陣列中的元素依循從零開始的編號系統，意即陣列中的第一個元素是元素 0，第二個元素是元素 1，依此類推。將元素括在中括號內 ("[]")。如果屬性選擇器指向陣列或物件，則篩選條件模式會與日誌格式不相符。如果 JSON 屬性包含句點 ("."")，則可以使用括號標記法來選取該屬性。

> **Note**
>
> 萬用字元選取器
> 您可以使用 JSON 萬用字元來選取任何陣列元素或任何 JSON 物件欄位。
> 配額
> 在屬性選取器中，您只能使用最多一個萬用字元選取器。

- Numeric operator (數值運算子)

  數值運算子以下列符號之一開始：大於 (">")、小於 ("<")、等於 ("=")、不等於 ("!=")、大於等於 (">=") 或是小於等於 ("<=")。

- 數字

  您可以使用包含加號 ("+") 或減號 ("-") 符號的整數，並遵守科學表示法。使用星號 ("*") 作為萬用字元來比對數字。

下列程式碼片段示範如何設定篩選條件模式的格式，使用數值來比對 JSON 詞彙。

```
// Filter pattern with greater than symbol
{ $.bandwidth > 75 }
// Filter pattern with less than symbol
{ $.latency < 50 }
// Filter pattern with greater than or equal to symbol
{ $.refreshRate >= 60 }
// Filter pattern with less than or equal to symbol
{ $.responseTime <= 5 }
// Filter pattern with equal sign
{ $.errorCode = 400}
// Filter pattern with not equal sign
{ $.errorCode != 500 }
// Filter pattern with scientific notation and plus symbol
{ $.number[0] = 1e+3 }
// Filter pattern with scientific notation and minus symbol
{ $.number[0] != 1e-3 }
```

使用簡單運算式來比對 JSON 日誌事件中的詞彙

下列範例中的程式碼片段，示範篩選條件模式如何比對 JSON 日誌事件中的詞彙。

> **Note**
>
> 如果要使用 JSON 日誌事件範例測試篩選條件模式範例，則必須在單行中輸入 JSON 日誌範例。

JSON 日誌事件

```
{
      "eventType": "UpdateTrail",
      "sourceIPAddress": "111.111.111.111",
      "arrayKey": [
            "value",
            "another value"
      ],
      "objectList": [
           {
             "name": "a",
             "id": 1
           },
           {
             "name": "b",
             "id": 2
           }
      ],
      "SomeObject": null,
      "cluster.name": "c"
}
```

Example: Filter pattern that matches string values

此篩選條件模式會比對屬性 "UpdateTrail" 中的字串 "eventType"。

```
{ $.eventType = "UpdateTrail" }
```

Example: Filter pattern that matches string values (IP address)

篩選條件模式包含一個萬用字元，並與屬性 "sourceIPAddress" 相符，因為其不包含帶有前綴的數字 "123.123."。

```
{ $.sourceIPAddress != 123.123.* }
```

Example: Filter pattern that matches a specific array element with a string value

此篩選條件模式會比對陣列 "value" 中的元素 "arrayKey"。

```
{ $.arrayKey[0] = "value" }
```

Example: Filter pattern that matches a string using regex

此篩選條件模式會比對屬性 "Trail" 中的字串 "eventType"。

```
{ $.eventType = %Trail% }
```

Example: Filter pattern that uses a wildcard to match values of any element in the array using regex

篩選條件模式包含可比對陣列 "arrayKey" 中元素 "value" 的 regex。

```
{ $.arrayKey[*] = %val.{2}% }
```

Example: Filter pattern that uses a wildcard to match values of any element with a specific prefix and subnet using regex (IP address)

此篩選條件模式包含可比對屬性 "sourceIPAddress" 中元素 "111.111.111.111" 的 regex。

```
{ $.* = %111\.111\.111\.1[0-9]{1,2}% }
```

> **Note**
>
> 配額

在屬性選取器中，您只能使用最多一個萬用字元選取器。

Example: Filter pattern that matches a JSON property with a period (.) in the key

```
{ $.['cluster.name'] = "c" }
```

Example: Filter pattern that matches JSON logs using IS

您可以使用 IS 變數建立比對 JSON 日誌中欄位的篩選條件模式。IS 變數可以比對包含 NULL、TRUE 或 FALSE 值的欄位。下列篩選條件模式會傳回 SomeObject 值為 NULL 的 JSON 日誌。

```
{ $.SomeObject IS NULL }
```

Example: Filter pattern that matches JSON logs using NOT EXISTS

您可以使用 NOT EXISTS變數建立篩選模式，以傳回日誌資料中不包含特定欄位的 JSON 日誌。下列篩選條件模式會使用 NOT EXISTS 傳回不包含欄位 SomeOtherObject 的 JSON 日誌。

```
{ $.SomeOtherObject NOT EXISTS }
```

Note

目前不支援變數 IS NOT 和 EXISTS。

使用複合運算式來比對 JSON 物件中的詞彙

您可以在篩選條件模式中使用邏輯運算子邏輯與 ("&&") 和邏輯或 ("||") 來建立兩個或多個條件為真的，與日誌事件相符的複合表達式。複合表達式支援使用小括號 ("()") 和以下標準運算次序：() > && > ||。下列範例中的程式碼片段，示範如何將篩選條件模式與複合表達式相結合，來比對 JSON 物件中的詞彙。

JSON 物件

```
{
    "user": {
        "id": 1,
        "email": "John.Stiles@example.com"
    },
    "users": [
        {
         "id": 2,
         "email": "John.Doe@example.com"
        },
        {
         "id": 3,
         "email": "Jane.Doe@example.com"
        }
    ],
    "actions": [
        "GET",
        "PUT",
        "DELETE"
    ],
    "coordinates": [
        [0, 1, 2],
        [4, 5, 6],
        [7, 8, 9]
    ]
}
```

Example: Expression that matches using AND (&&)

此篩選條件模式包含的複合表達式，將 "user" 中的 "id" 與數值 1，和 "users" 陣列中第一個元素的 "email" 與字串 "John.Doe@example.com" 做比對。

```
{ ($.user.id = 1) && ($.users[0].email = "John.Doe@example.com") }
```

Example: Expression that matches using OR (||)

此篩選條件模式包含的複合表達式，將 "user" 中的 "email" 與字串 "John.Stiles@example.com" 相比對。

```
{ $.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch" &&
 $.actions[2] = "nonmatch" }
```

Example: Expression that doesn't match using AND (&&)

此篩選條件模式包含一個找不到相符項目的複合表達式，因為該表達式與 "actions" 中的第三個動作不相符。

```
{ ($.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch") &&
 $.actions[2] = "nonmatch" }
```

> **Note**
>
> 配額
> 在屬性選取器中，您只能使用最多一個萬用字元選取器，在具有複合運算式的篩選條件模式中，只能使用最多三個萬用字元選取器。

Example: Expression that doesn't match using OR (||)

此篩選條件模式包含一個找不到相符項目的複合表達式，因為該表達式與 "users" 中的第一個屬性，或 "actions" 中的第三個動作不符。

```
{ ($.user.id = 2 && $.users[0].email = "nonmatch") || $.actions[2] = "GET" }
```

使用篩選條件模式來比對以空格分隔的日誌事件中的詞彙

編寫以空格分隔的日誌事件的篩選條件模式

您可以建立篩選條件模式來比對以空格分隔的日誌事件中的詞彙。下面提供了以空格分隔的日誌事件範例，示範了如何編寫用於比對空格分隔日誌事件中詞彙的篩選條件模式語法。

> **Note**
>
> 建立篩選條件模式來比對以空格分隔的日誌事件中的詞彙時，您可以使用任何條件式規則運算式。如需支援的規則運算式清單，請參閱支援的規則運算式。

Example: Space-delimited log event

以下程式碼片段為以空格分隔的日誌事件，其中包含七個欄位：`ip`、`user`、`username`、`timestamp`、`request`、`status_code`，和 `bytes`。

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404
 1534
```

> **Note**
>
> 中括號 ("[]") 和雙引號 ("") 之間的字元被視為單一欄位。

Writing filter patterns that match terms in a space-delimited log event

若要建立用來比對以空格分隔的日誌事件中詞彙的篩選條件模式，須將篩選條件模式括在中括號 ("[]") 中，並指定逗點 (",") 分隔名稱的欄位。下列篩選條件模式會分析七個欄位。

```
[ip=%127\.0\.0\.[1-9]%, user, username, timestamp, request =*.html*, status_code =
 4*, bytes]
```

您可以使用數字運算子 (>、<、=、!=、>= 或 <=) 和星號 (*) 作為萬用字元或 regex，來建立篩選條件模式。在範例篩選條件模式中，ip 使用可比對 IP 地址範圍 127.0.0.1 - 127.0.0.9 的 regex，其中 request 包含萬用字元，說明必須擷取包含 .html 的值，status_code 包含萬用字元，說明必須擷取以 4 開頭的值。

如果您不知道在空格分隔的日誌事件中剖析的欄位數量，則可以使用刪節號 (...) 來引用任何未命名的欄位。刪節號可以用來引用所需數量的欄位。下列範例為帶刪節號的篩選條件模式，代表先前範例篩選條件模式中的前四個未命名欄位。

```
[..., request =*.html*, status_code = 4*, bytes]
```

您也可以使用邏輯運算子，邏輯與 (&&) 和邏輯或 (||) 來建立複合表達式。下列篩選條件模式包含一個複合表達式，該表達式表明 status_code 的值必須是 404 或是 410。

```
[ip, user, username, timestamp, request =*.html*, status_code = 404 || status_code =
 410, bytes]
```

使用模式比對來比對以空格分隔的日誌事件中的詞彙

您可以使用模式比對，來建立按照特定順序比對詞彙的以空格分隔的篩選條件模式。使用指示器指定詞彙的順序。使用 w1 來表示第一個詞彙，並使用 w2 等來表示後續詞彙的順序。在詞彙之間插入逗號(",")。下列範例中的程式碼片段示範了如何使用模式，來比對以空格分隔的篩選條件模式。

> **Note**
>
> 建立篩選條件模式來比對以空格分隔的日誌事件中的詞彙時，您可以使用任何條件式規則運算式。如需支援的規則運算式清單，請參閱支援的規則運算式。

以空格分隔的日誌事件

```
INFO 09/25/2014 12:00:00 GET /service/resource/67 1200
INFO 09/25/2014 12:00:01 POST /service/resource/67/part/111 1310
```

```
WARNING 09/25/2014 12:00:02 Invalid user request
ERROR 09/25/2014 12:00:02 Failed to process request
```

Example: Match terms in order

下列以空格分隔的篩選條件模式會傳回第一個單字為 ERROR (錯誤) 的日誌事件。

```
[w1=ERROR, w2]
```

> **Note**
>
> 建立使用模式比對的以空格分隔的篩選條件模式時，必須在指定詞彙的順序後加上一個空白指標。例如：如果您建立了一個篩選條件模式以傳回第一個單字為 ERROR 的日誌事件，請在 w1 詞彙後加上一個空白 w2 指標。

Example: Match terms with AND (&&) and OR (||)

您可以使用邏輯運算子邏輯與 ("&&") 和邏輯或 ("||") 建立包含條件的以空格分隔的篩選條件模式。下列篩選條件模式會傳回第一個單字為 ERROR (錯誤) 或 WARNING (警告) 的日誌事件。

```
[w1=ERROR || w1=WARNING, w2]
```

Example: Exclude terms from matches

您可以建立以空格分隔的篩選條件模式，以傳回排除一個或多個詞彙的日誌事件。將不等於符號 ("!=") 放在想要排除的一個或多個詞彙之前。下列程式碼片段為篩選條件模式範例，其會傳回第一個單字不是 ERROR (錯誤) 和 WARNING (警告) 的日誌事件。

```
[w1!=ERROR && w1!=WARNING, w2]
```

Example: Match the top level item in a resource URI

下列程式碼片段為篩選條件模式的範例，其使用 regex 比對資源 URI 中的上層詞彙。

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+$%, response_time]
```

Example: Match the child level item in a resource URI

下列程式碼片段為篩選條件模式的範例，其使用 regex 比對資源 URI 中的子層詞彙。

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+/part/[0-9]+$%,
 response_time]
```

從 AWS 服務啟用記錄

雖然許多服務只會將日誌發佈至 CloudWatch Logs，但某些 AWS 服務可以直接將日誌發佈至 Amazon Simple Storage Service 或 Amazon Data Firehose。如果您的日誌主要需求是在這些服務之一中進行儲存或處理，您可以輕鬆讓產生日誌的服務直接傳送到 Amazon S3 或 Firehose，而無需額外設定。

即使您將日誌直接發佈至 Amazon S3 或 Firehose，也會收取 CloudWatch 交付費用。如果您將日誌傳送至 Amazon S3，則*AWS_REGION*-S3-Egress-Bytes費用會顯示在 Cost Explorer 或帳單上。如果您將日誌傳送至 Firehose，則會顯示*AWS_REGION*-FH-Egress-Bytes費用。如需付費日誌定價的詳細資訊，請參閱 Amazon CloudWatch 定價中的日誌索引標籤。

有些 AWS 服務使用常見的基礎設施來傳送其日誌。若要啟用從這些服務記錄日誌，您必須以具有特定許可的使用者身分登入。此外，您必須授予 許可 AWS ，才能傳送日誌。

對於需要這些許可的服務，需要兩種許可版本。在資料表中會將需要這些額外許可的服務標註為支援的 [V1 許可] 和 支援的 [V2 許可] 。如需有關這些必要許可的詳細資訊，請參閱資料表後面的章節。

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
Amazon API Gateway 存取日誌	付費日誌	支援的 [V1 許可]			
AWS AppSync logs	自訂日誌	支援			
Amazon Aurora MySQL 日誌	自訂日誌	支援			
Amazon Bedrock 知識庫記錄	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Bedrock 客服人員記錄	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
Amazon Bedrock AgentCore 執行期	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]
Amazon Bedrock AgentCore 閘道	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]
Amazon Bedrock AgentCore Identity	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]
Amazon Bedrock AgentCore 記憶體	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]
Amazon Bedrock AgentCore 工具	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]
Amazon Chime 媒體品質指標日誌和 SIP 訊息日誌	付費日誌	支援的 [V1 許可]			
CloudFront：存取日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS CloudHSM 稽核日誌	自訂日誌	支援			
CloudWatch Evidently 評估事件日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]		
CloudWatch 網路監視器日誌	付費日誌		支援的 [V1 許可]		
CloudTrail 日誌	自訂日誌	支援			

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
AWS CodeBuild logs	自訂日誌	支援			
Amazon CodeWhisperer 事件日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Cognito logs	付費日誌	支援的 [V1 許可]			
Amazon Connect 日誌	自訂日誌	支援			
AWS DataSync logs	自訂日誌	支援			
AWS DevOps Agent logs	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon ElastiCache (Redis OSS) 日誌	付費日誌	支援的 [V1 許可]		支援的 [V1 許可]	
AWS Elastic Beanstalk 日誌	自訂日誌	支援			
Amazon Elastic Container Service 日誌	自訂日誌	支援			
Amazon Elastic Kubernetes Service Auto 模式日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Elastic Kubernetes Service 控制平面日誌	付費日誌	支援			
AWS Elemental MediaPackage 存取日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
AWS Elemental MediaTailor logs	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS Entity Resolution 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon EventBridge 管道記錄	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon EventBridge 事件匯流排	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS Fargate 日誌	自訂日誌	支援			
AWS Fault Injection Service 實驗日誌	付費日誌		支援的 [V1 許可]		
Amazon FinSpace	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
AWS Global Accelerator 流程日誌	付費日誌		支援的 [V1 許可]		
AWS Glue 工作日誌	自訂日誌	支援			
IAM Identity Center 錯誤日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Interactive Video Service 聊天日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
AWS IoT 日誌	自訂日誌	支援			
AWS IoT FleetWise logs	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
AWS Lambda 日誌	付費日誌	支援	支援	支援	
Amazon Macie 日誌	自訂日誌	支援			
Amazon SES 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS Mainframe Modernization	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon Managed Service for Prometheus	付費日誌	支援的 [V1 許可]			
Amazon MSK 代理程式日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon MSK Connect 日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon MQ 日誌	自訂日誌	支援			
AWS Network Firewall 日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
AWS Network Firewall Proxy 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
Network Load Balancer 存取日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
OpenSearch 日誌	自訂日誌	支援			
Amazon OpenSearch Service 擷取日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
AWS PCS 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Q Business 連接器日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Q Business 對話日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Quick Chat 和意見回饋日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon 關聯式資料庫 ServicePostgreSQL 日誌	自訂日誌	支援			
AWS RTB Fabric 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS Security Hub CSPM	付費日誌	支援的 [V2 許可]			
Amazon Route 53 公有 DNS 查詢日誌	付費日誌	支援			

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
Amazon Route 53 Resolver 查詢日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]		
Amazon SageMaker AI 事件	付費日誌	支援的 [V1 許可]			
Amazon SageMaker AI 工作者事件	付費日誌	支援的 [V1 許可]			
AWS Site-to_Site VPN 日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon Simple Email Service 日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
Amazon Simple Notification Service 日誌	自訂日誌	支援			
Amazon Simple Notification Service 資料保護政策日誌	自訂日誌	支援			
EC2 Spot 執行個體資料摘要檔案	付費日誌		支援的 [V1 許可]		
AWS Step Functions 快速工作流程和標準工作流程日誌	付費日誌	支援的 [V1 許可]			
Storage Gateway 稽核日誌和運作狀態日誌	付費日誌	支援的 [V1 許可]			

日誌來源	日誌類型	the section called “傳送至 CloudWatch Logs 的日誌”	the section called “傳送至 Amazon S3 的日誌”	the section called “傳送至 Firehose 的日誌”	the section called “傳送至 X-Ray 的追蹤”
AWS Transfer Family logs	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
AWS Verified Access logs	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon Virtual Private Cloud 流程日誌	付費日誌	支援	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon VPC Lattice 存取日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援的 [V1 許可]	
Amazon VPC Route Server	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	
AWS WAF 日誌	付費日誌	支援的 [V1 許可]	支援的 [V1 許可]	支援	
Amazon WorkMail 稽核日誌	付費日誌	支援的 [V2 許可]	支援的 [V2 許可]	支援的 [V2 許可]	

需要額外許可 [V1] 的日誌記錄

有些 AWS 服務會使用常見的基礎設施，將日誌傳送至 CloudWatch Logs、Amazon S3 或 Firehose。若要讓上表中列出的 AWS 服務將日誌傳送至這些目的地，您必須以具有特定許可的使用者身分登入。

此外，必須授予 許可 AWS ，才能傳送日誌。 AWS 可以在設定日誌時自動建立這些許可，或者您可以在設定日誌之前先自行建立這些許可。若要跨帳戶交付，您必須自行手動建立許可政策。

如果您選擇在您或組織中的某人第一次設定日誌傳送時，讓 AWS 自動設定必要的許可和資源政策，則設定日誌傳送的使用者必須具有特定許可，如本節稍後所述。或者，您可以自行建立資源政策，所以設定傳送日誌的使用者就不需要這麼多許可。

下列主題提供每個目的地的更多詳細資訊。

主題

傳送至 CloudWatch Logs 的日誌

> **Important**
>
> 當您將下列清單中的日誌類型設定為傳送至 CloudWatch Logs 時， AWS 會視需要為接收日誌的日誌群組建立或變更相關聯的資源政策。繼續閱讀本節以查看詳細資訊。

本節適用於將上一節表中列出的日誌類型，傳送至 CloudWatch Logs 的情況：

使用者許可

您必須以具有下列許可的帳戶登入，才能第一次設定將任何這些類型的日誌傳送到 CloudWatch Logs。

- `logs:CreateLogDelivery`
- `logs:PutResourcePolicy`
- `logs:DescribeResourcePolicies`
- `logs:DescribeLogGroups`

> **Note**
>
> 當您指定 `logs:DescribeLogGroups`、 `logs:DescribeResourcePolicies`或 `logs:PutResourcePolicy`許可時，請務必將其`Resource`行的 ARN 設定為使用`*`萬用字元，而不是只指定單一日誌群組名稱。例如 `"Resource": "arn:aws:logs:us-east-1:111122223333:log-group:*"`

如果任何這些類型的日誌已傳送到 CloudWatch Logs 中的某個日誌群組，則若要設定將另一種類型的日誌傳送到同一個日誌群組，您只需要 logs:CreateLogDelivery 許可。

日誌群組和資源政策

日誌送往的日誌群組必須具有包含特定許可的資源政策。如果日誌群組目前沒有資源政策，且設定記錄的使用者具有日誌群組的 logs:PutResourcePolicy、logs:DescribeResourcePolicies 及logs:DescribeLogGroups 許可，則當您開始將日誌傳送至 CloudWatch Logs 時， AWS 會自動建立下列政策。對於新建立的訂閱，資源政策是在日誌群組層級設定，大小上限為 51，200 個位元組。如果現有的帳戶層級資源政策已透過萬用字元授予許可，則不會建立單獨的日誌群組層級政策。若要檢查特定日誌群組的 logGroup 層級資源政策，請使用 describe-resource-policies命令，並將 --resource-arn 參數設定為日誌群組 ARN，並將 --policy-scope 參數設定為 RESOURCE。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "delivery.logs.amazonaws.com"
                ]
            },
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group:log-
stream:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
```

```
                    "arn:aws:logs:us-east-1:111122223333:*"
                ]
            }
        }
    }
]
}
```

日誌群組的資源政策限制為 51，200 個位元組。一旦達到此限制，AWS 就無法新增新許可。這需要客戶手動修改政策，以授予 logs:CreateLogStream和 logs:PutLogEvents動作delivery.logs.amazonaws.com的服務主體許可。客戶應該將日誌群組名稱字首與萬用字元搭配使用，例如 ，/aws/vendedlogs/*並將此日誌群組名稱用於未來的交付建立。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "delivery.logs.amazonaws.com"
                ]
            },
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group/aws/
vendedlogs/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
```

```
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:*"
                    ]
                }
            }
        }
    ]
}
```

傳送至 Amazon S3 的日誌

當您將日誌設定為傳送至 Amazon S3 時， 會視需要 AWS 建立或變更與接收日誌的 S3 儲存貯體相關聯的資源政策。

直接發佈至 Amazon S3 的日誌會發佈至您指定的現有儲存貯體。在指定的儲存貯體中，每五分鐘會建立一或多個日誌檔案。

當您第一次將日誌傳送到 Amazon S3 儲存貯體時，傳送日誌的服務會記錄儲存貯體的擁有者，以確保日誌僅傳送到屬於此帳戶的儲存貯體。因此，若要變更 Amazon S3 儲存貯體擁有者，您必須在原始服務中重新建立或更新日誌訂閱。

> **Note**
>
> CloudFront 使用的許可模型與其他將付費日誌傳送至 S3 的 服務不同。如需詳細資訊，請參閱設定標準記錄和存取日誌檔案所需的許可。
> 此外，如果您針對 CloudFront 存取日誌和另一個日誌來源使用相同的 S3 儲存貯體，在 CloudFront 的儲存貯體上啟用 ACL 也會將許可授予使用此儲存貯體的所有其他日誌來源。

> **Important**
>
> 如果您要將日誌傳送至 Amazon S3 儲存貯體，且儲存貯體政策包含 `NotAction`或 `NotPrincipal`元素，則自動將日誌交付許可新增至儲存貯體，並建立日誌訂閱將會失敗。若要成功建立日誌訂閱，您需要手動將日誌交付許可新增至儲存貯體政策，然後建立日誌訂閱。如需詳細資訊，請參閱本節中的指示。
> 如果儲存貯體使用客戶受管 AWS KMS 金鑰進行伺服器端加密，您還必須為客戶受管金鑰新增金鑰政策。如需詳細資訊，請參閱Amazon S3。

如果目的地儲存貯體已啟用 SSE-KMS 和儲存貯體金鑰，則連接的客戶受管 KMS 金鑰政策不再適用於所有請求。如需詳細資訊，請參閱使用 Amazon S3 儲存貯體金鑰降低 SSE-KMS 的成本。
如果您使用付費日誌和 S3 加密搭配客戶受 AWS KMS 管金鑰，則必須在設定儲存貯體時使用完整 AWS KMS 金鑰 ARN，而非金鑰 ID。如需詳細資訊，請參閱 put-bucket-encryption。

使用者許可

您必須以具有下列許可的帳戶登入，才能第一次設定將任何這些類型的日誌傳送到 Amazon S3。

- `logs:CreateLogDelivery`
- `S3:GetBucketPolicy`
- `S3:PutBucketPolicy`

如果任何這些類型的日誌已傳送到某個 Amazon S3 儲存貯體，則若要設定將另一種類型的日誌傳送到同一個儲存貯體，您只需要有 `logs:CreateLogDelivery` 許可。

S3 儲存貯體資源政策

日誌送往的 S3 儲存貯體必須具有包含特定許可的資源政策。如果儲存貯體目前沒有資源政策，且設定記錄的使用者具有儲存貯體的 `S3:GetBucketPolicy` 和 `S3:PutBucketPolicy` 許可，則當您開始將日誌傳送至 Amazon S3 時， AWS 會自動建立下列政策。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "AWSLogDeliveryWrite20150319",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryAclCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:*"
                    ]
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/account-ID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:*"
                    ]
                }
            }
        }
    ]
}
```

在先前的政策中，對於 aws:SourceAccount，指定要將日誌交付至此儲存貯體的帳戶 IDS 清單。對於 aws:SourceArn，指定產生日誌之資源的 ARN 清單，格式為 arn:aws:logs:*source-region*:*source-account-id*:*。

如果儲存貯體具有資源政策，但該政策未包含前一個政策中出現的陳述式，且設定記錄的使用者具有儲存貯體的 S3:GetBucketPolicy 和 S3:PutBucketPolicy 許可，則該陳述式會附加至儲存貯體的資源政策。

> **Note**
>
> 在某些情況下， AWS CloudTrail 如果 s3:ListBucket許可尚未授予 ，您可能會在 中看到AccessDenied錯誤delivery.logs.amazonaws.com。若要避免 CloudTrail 日誌中出現這些錯誤，您必須將 s3:ListBucket 許可授予 delivery.logs.amazonaws.com，且必須包含與先前儲存貯體政策中設定的 s3:GetBucketAcl 許可一起顯示的 Condition 參數。為簡化此操作而不用建立一個新的 Statement，你可以直接將 AWSLogDeliveryAclCheck 更新為 “Action”: [“s3:GetBucketAcl”, “s3:ListBucket”]

Amazon S3 儲存貯體伺服器端加密

您可以啟用伺服器端加密搭配 Amazon S3 S3-managed金鑰 (SSE-S3) 或使用存放在 AWS Key Management Service (SSE-KMS) 中的 AWS KMS 金鑰進行伺服器端加密，以保護 Amazon S3 儲存貯體中的資料。如需詳細資訊，請參閱使用伺服器端加密保護資料。

如果您選擇 SSE-S3，則不需要其他組態。Amazon S3 會處理加密金鑰。

> **Warning**
>
> 如果您選擇 SSE-KMS，您必須使用客戶受管金鑰，因為此案例不支援使用 AWS 受管金鑰。如果您使用 AWS 受管金鑰設定加密，日誌將以無法讀取的格式交付。

當您使用客戶受管 AWS KMS 金鑰時，您可以在啟用儲存貯體加密時指定客戶受管金鑰的 Amazon Resource Name (ARN)。您必須將以下內容新增至客戶受管金鑰的金鑰政策 (而不是 S3 儲存貯體的儲存貯體政策)，以便日誌傳遞帳戶可以寫入您的 S3 儲存貯體。

如果您選擇 SSE-KMS，則必須使用客戶受管金鑰，因為此情況不支援使用 AWS 受管金鑰。當您使用客戶受管 AWS KMS 金鑰時，您可以在啟用儲存貯體加密時指定客戶受管金鑰的 Amazon Resource Name (ARN)。您必須將以下內容新增至客戶受管金鑰的金鑰政策 (而不是 S3 儲存貯體的儲存貯體政策)，以便日誌傳遞帳戶可以寫入您的 S3 儲存貯體。

```
{
```

```
    "Sid": "Allow Logs Delivery to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": [ "delivery.logs.amazonaws.com" ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": ["0123456789"]
        },
        "ArnLike": {
            "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:*"]
        }
        }
}
```

對於 aws:SourceAccount，指定要將日誌交付至此儲存貯體的帳戶 IDS 清單。對於 aws:SourceArn，指定產生日誌之資源的 ARN 清單，格式為 arn:aws:logs:*source-region*:*source-account-id*:*。

傳送至 Firehose 的日誌

本節適用於上一節資料表中列出的日誌類型傳送至 Firehose 的情況：

使用者許可

若要第一次設定將任何這類日誌傳送至 Firehose，您必須使用下列許可登入 帳戶。

- logs:CreateLogDelivery
- firehose:TagDeliveryStream
- iam:CreateServiceLinkedRole

如果這些類型的日誌中有任何一種已經傳送到 Firehose，則若要設定將另外一種類型的日誌傳送至 Firehose，您只需要擁有 logs:CreateLogDelivery和 firehose:TagDeliveryStream許可。

用於許可的 IAM 角色

由於 Firehose AWS 不使用資源政策，因此 會在設定將這些日誌傳送至 Firehose 時使用 IAM 角色。會 AWS 建立名為 的服務連結角色AWSServiceRoleForLogDelivery。此服務連結角色包含下列許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

此服務連結角色會針對LogDeliveryEnabled標籤設定為 的所有 Firehose 交付串流授予許可true。當您設定記錄時，會將此標籤 AWS 提供給目的地交付串流。

此服務連結角色也有信任政策，以允許 delivery.logs.amazonaws.com 服務委託人擔任所需的服務連結角色。該信任政策如下：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

需要額外許可 [V2] 的日誌記錄

有些 AWS 服務使用新方法來傳送其日誌。這是一種靈活的方法，可讓您設定從這些服務到下列一或多個目的地的日誌交付：CloudWatch Logs、Amazon S3 或 Firehose 和 X-Ray 以進行追蹤交付。

工作日誌交付包含三個元素：

- `DeliverySource`（一種）是邏輯物件，代表實際傳送日誌的資源。
- `DeliveryDestination`，這是代表實際交付目的地的邏輯物件。
- `Delivery`將交付來源連線至交付目的地的

若要設定支援 AWS 的服務與目的地之間的日誌交付，您必須執行下列動作：

- 使用 PutDeliverySource 建立交付來源。
- 使用 PutDeliveryDestination 建立交付目的地。
- 如果您要跨帳戶交付日誌，則必須在目的地帳戶中使用 PutDeliveryDestinationPolicy 將 IAM 政策指派給目的地。此政策授權從帳戶 A 中的交付來源建立交付至帳戶 B 中的交付目的地。若要跨帳戶交付，您必須自行手動建立許可政策。
- 使用 CreateDelivery。

以下各節提供您在登入後使用 V2 處理程序設定每種目的地的日誌傳遞所需之許可的詳細資訊。可將這些許可授予您登入時具有的 IAM 角色。

> **Important**
>
> 刪除日誌產生資源後，您有責任移除日誌交付資源。若要這樣做，請遵循下列步驟。
>
> 1. `Delivery` 使用 DeleteDelivery 操作刪除 。
> 2. `DeliverySource` 使用 DeleteDeliverySource 操作刪除 。

3. 如果與您剛刪除DeliverySource的 DeliveryDestination相關聯的 僅用於此特定 DeliverySource，則您可以使用 DeleteDeliveryDestinations 操作將其移除。

內容

傳送至 CloudWatch Logs 的日誌

使用者許可

若要啟用傳送日誌至 CloudWatch Logs，您登入時必須具有以下許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:GetDelivery",
                "logs:GetDeliverySource",
                "logs:PutDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:DeleteDeliverySource",
                "logs:PutDeliveryDestinationPolicy",
                "logs:CreateDelivery",
                "logs:GetDeliveryDestination",
                "logs:PutDeliverySource",
                "logs:DeleteDeliveryDestination",
                "logs:DeleteDeliveryDestinationPolicy",
                "logs:DeleteDelivery",
```

```
                "logs:UpdateDeliveryConfiguration"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:111122223333:delivery:*",
            "arn:aws:logs:us-east-1:444455556666:delivery-source:*",
            "arn:aws:logs:us-east-1:777788889999:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries",
                "logs:DescribeConfigurationTemplates"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyCWL",
            "Effect": "Allow",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:123456789012:*"
            ]
        }
    ]
}
```

日誌群組和資源政策

日誌送往的日誌群組必須具有包含特定許可的資源政策。如果日誌群組目前沒有資源政策，且設定記錄的使用者具有日誌群組的 `logs:PutResourcePolicy`、`logs:DescribeResourcePolicies` 及`logs:DescribeLogGroups` 許可，則當您開始將日誌傳送至 CloudWatch Logs 時， AWS 會自動建立下列政策。對於新建立的訂閱，資源政策是在日誌群組層級設定，大小上限為 51，200 個位元組。如果現有的帳戶層級資源政策已透過萬用字元授予許可，則不會建立單獨的日誌群組層級政策。若

要檢查特定日誌群組的 logGroup 層級資源政策，請使用 `describe-resource-policies`命令，並將 `--resource-arn` 參數設定為日誌群組 ARN，並將 `--policy-scope` 參數設定為 `RESOURCE`。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "delivery.logs.amazonaws.com"
                ]
            },
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group:log-
stream:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:*"
                    ]
                }
            }
        }
    ]
}
```

日誌群組的資源政策限制為 51，200 個位元組。一旦達到此限制，AWS 就無法新增新許可。這需要客戶手動修改政策，以授予 logs:CreateLogStream和 logs:PutLogEvents動作delivery.logs.amazonaws.com的服務主體許可。客戶應該將日誌群組名稱字首與萬用字元搭配使用，例如 ，/aws/vendedlogs/*並將此日誌群組名稱用於未來的交付建立。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "delivery.logs.amazonaws.com"
                ]
            },
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group/aws/
vendedlogs/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:*"
                    ]
                }
            }
        }
    ]
}
```

傳送至 Amazon S3 的日誌

使用者許可

若要啟用傳送日誌至 Amazon S3，您登入時必須具有以下許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:GetDelivery",
                "logs:GetDeliverySource",
                "logs:PutDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:DeleteDeliverySource",
                "logs:PutDeliveryDestinationPolicy",
                "logs:CreateDelivery",
                "logs:GetDeliveryDestination",
                "logs:PutDeliverySource",
                "logs:DeleteDeliveryDestination",
                "logs:DeleteDeliveryDestinationPolicy",
                "logs:DeleteDelivery",
                "logs:UpdateDeliveryConfiguration"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:111122223333:delivery:*",
    "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
    "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries",
                "logs:DescribeConfigurationTemplates"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyS3",
            "Effect": "Allow",
            "Action": [
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
            "Resource": "arn:aws:s3:::bucket-name"
        }
    ]
}
```

日誌送往的 S3 儲存貯體必須具有包含特定許可的資源政策。如果儲存貯體目前沒有資源政策，且設定記錄的使用者具有儲存貯體的 S3:GetBucketPolicy 和 S3:PutBucketPolicy 許可，則當您開始將日誌傳送至 Amazon S3 時， AWS 會自動建立下列政策。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "AWSLogDeliveryWrite20150319",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/account-ID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": [
                        "0123456789"
                    ]
                },
                "ArnLike": {
```

```
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:111122223333:delivery-source:*"
                    ]
                }
            }
        }
    ]
}
```

在先前的政策中，對於 aws:SourceAccount，指定要將日誌交付至此儲存貯體的帳戶 IDS 清單。對於 aws:SourceArn，指定產生日誌之資源的 ARN 清單，格式為 arn:aws:logs:*source-region*:*source-account-id*:*。

如果儲存貯體具有資源政策，但該政策未包含前一個政策中出現的陳述式，且設定記錄的使用者具有儲存貯體的 S3:GetBucketPolicy 和 S3:PutBucketPolicy 許可，則該陳述式會附加至儲存貯體的資源政策。

> **Note**
>
> 在某些情況下， AWS CloudTrail 如果 s3:ListBucket許可尚未授予 ，您可能會在 中看到AccessDenied錯誤delivery.logs.amazonaws.com。若要避免 CloudTrail 日誌中出現這些錯誤，您必須將 s3:ListBucket 許可授予 delivery.logs.amazonaws.com，且必須包含與先前儲存貯體政策中設定的 s3:GetBucketAcl 許可一起顯示的 Condition 參數。為簡化此操作而不用建立一個新的 Statement，你可以直接將 AWSLogDeliveryAclCheck 更新為 “Action”: [“s3:GetBucketAcl”, “s3:ListBucket”]

Amazon S3 儲存貯體伺服器端加密

您可以啟用伺服器端加密搭配 Amazon S3 S3-managed金鑰 (SSE-S3) 或使用存放在 AWS Key Management Service (SSE-KMS) 中的 AWS KMS 金鑰進行伺服器端加密，以保護 Amazon S3 儲存貯體中的資料。如需詳細資訊，請參閱使用伺服器端加密保護資料。

如果您選擇 SSE-S3，則不需要其他組態。Amazon S3 會處理加密金鑰。

> **Warning**
>
> 如果您選擇 SSE-KMS，您必須使用客戶受管金鑰，因為此案例不支援使用 AWS 受管金鑰。如果您使用 AWS 受管金鑰設定加密，日誌將以無法讀取的格式交付。

當您使用客戶受管 AWS KMS 金鑰時，您可以在啟用儲存貯體加密時指定客戶受管金鑰的 Amazon Resource Name (ARN)。您必須將以下內容新增至客戶受管金鑰的金鑰政策 (而不是 S3 儲存貯體的儲存貯體政策)，以便日誌傳遞帳戶可以寫入您的 S3 儲存貯體。

如果您選擇 SSE-KMS，則必須使用客戶受管金鑰，因為此情況不支援使用 AWS 受管金鑰。當您使用客戶受管 AWS KMS 金鑰時，您可以在啟用儲存貯體加密時指定客戶受管金鑰的 Amazon Resource Name (ARN)。您必須將以下內容新增至客戶受管金鑰的金鑰政策 (而不是 S3 儲存貯體的儲存貯體政策)，以便日誌傳遞帳戶可以寫入您的 S3 儲存貯體。

```
{
    "Sid": "Allow Logs Delivery to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": [ "delivery.logs.amazonaws.com" ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": ["0123456789"]
        },
        "ArnLike": {
            "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:delivery-source:*"]
        }
        }
}
```

對於 aws:SourceAccount，指定要將日誌交付至此儲存貯體的帳戶 IDS 清單。對於 aws:SourceArn，指定產生日誌之資源的 ARN 清單，格式為 arn:aws:logs:*source-region*:*source-account-id*:*。

傳送至 Firehose 的日誌

使用者許可

若要啟用傳送日誌至 Firehose，您必須使用以下許可登入。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:GetDelivery",
                "logs:GetDeliverySource",
                "logs:PutDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:DeleteDeliverySource",
                "logs:PutDeliveryDestinationPolicy",
                "logs:CreateDelivery",
                "logs:GetDeliveryDestination",
                "logs:PutDeliverySource",
                "logs:DeleteDeliveryDestination",
                "logs:DeleteDeliveryDestinationPolicy",
                "logs:DeleteDelivery",
                "logs:UpdateDeliveryConfiguration"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:111122223333:delivery:*",
    "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
    "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
```

```
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries",
                "logs:DescribeConfigurationTemplates"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyFH",
            "Effect": "Allow",
            "Action": [
                "firehose:TagDeliveryStream"
            ],
            "Resource": [
            "arn:aws:firehose:us-east-1:111122223333:deliverystream/*"
            ]
        },
        {
            "Sid": "CreateServiceLinkedRole",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery"
        }
    ]
}
```

用於資源許可的 IAM 角色

由於 Firehose AWS 不使用資源政策，因此 會在設定將這些日誌傳送至 Firehose 時使用 IAM 角色。會 AWS 建立名為 的服務連結角色AWSServiceRoleForLogDelivery。此服務連結角色包含下列許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
```

```
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

此服務連結角色會針對LogDeliveryEnabled標籤設定為 的所有 Firehose 交付串流授予許可true。當您設定記錄時，會將此標籤 AWS 提供給目的地交付串流。

此服務連結角色也有信任政策，以允許 delivery.logs.amazonaws.com 服務委託人擔任所需的服務連結角色。該信任政策如下：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

傳送至 X-Ray 的追蹤

使用者許可

若要啟用傳送追蹤到 AWS X-Ray，您必須使用下列許可登入。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccessForLogDeliveryActions",
            "Effect": "Allow",
              "Action": [
                "logs:GetDelivery",
                "logs:GetDeliverySource",
                "logs:PutDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:DeleteDeliverySource",
                "logs:PutDeliveryDestinationPolicy",
                "logs:CreateDelivery",
                "logs:GetDeliveryDestination",
                "logs:PutDeliverySource",
                "logs:DeleteDeliveryDestination",
                "logs:DeleteDeliveryDestinationPolicy",
                "logs:DeleteDelivery",
                "logs:UpdateDeliveryConfiguration"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:111122223333:delivery:*",
            "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
            "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries",
                "logs:DescribeConfigurationTemplates"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "AllowUpdatesToResourcePolicyXRay",
            "Effect": "Allow",
            "Action": [
                "xray:PutResourcePolicy",
                "xray:ListResourcePolicies",
                "xray:GetTraceSegmentDestination"
            ],
            "Resource": "*"
        }
    ]
}
```

X-Ray 資源政策

正在傳送追蹤的目的地帳戶必須具有包含特定許可的資源政策。當設定追蹤的使用者在帳戶中具有 `xray:PutResourcePolicy`和 `xray:ListResourcePolicies`許可時，當您開始將追蹤傳送至 X-Ray 時 AWS ， 會自動建立資源政策。建立的政策取決於來源服務 ：

Amazon Bedrock AgentCore 資源

AWS 會為每個資源類型建立一個資源政策。此政策使用範圍為帳戶界限的萬用字元模式，涵蓋帳戶中相同 Amazon Bedrock AgentCore 資源類型的所有資源。例如，如果啟用 Amazon Bedrock AgentCore 記憶體資源進行追蹤交付，政策會涵蓋該帳戶中的所有記憶體資源，包括未來建立的任何記憶體資源。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "xray:PutTraceSegments",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
```

```
        "ForAllValues:ArnLike": {
          "logs:LogGeneratingResourceArns": "arn:aws:bedrock-agentcore:us-
east-1:123456789012:memory/*"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:logs:us-east-1:123456789012:delivery-
source:*"
        }
      }
    }
  ]
}
```

AWS 其他服務

對於支援追蹤交付的其他 服務，AWS 會建立範圍限定於特定來源資源的資源政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "xray:PutTraceSegments",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ForAllValues:ArnLike": {
          "logs:LogGeneratingResourceArns": "arn:aws:bedrock:us-
east-1:123456789012:knowledge-base/KnowledgeBaseId"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:logs:us-east-1:123456789012:delivery-
source:xray-test"
        }
      }
    }
```

```
  ]
}
```

啟用交易搜尋

若要啟用傳送追蹤到 X-Ray，您必須啟用交易搜尋。

服務特定許可

除了前面各節中列出的目的地特定許可之外，某些服務需要明確授權，允許客戶從其資源傳送日誌，作為額外的安全層。它會針對在該服務內提供日誌的資源授權 `AllowVendedLogDeliveryForResource`動作。對於這些服務，請使用下列政策，並以適當的值取代*#######*。如需這些欄位的服務特定值，請參閱這些服務的文件頁面以取得付費日誌。在下列範例中，政策已更新，以啟用來自 Amazon SES 的付費日誌。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceLevelAccessForLogDelivery",
            "Effect": "Allow",
            "Action": [
                "ses:AllowVendedLogDeliveryForResource"
            ],
            "Resource": "arn:aws:ses:us-east-1:123456789012:resource-type/*"
        }
    ]
}
```

主控台特定許可

除了先前章節中列出的許可之外，如果您使用主控台而非 APIs 設定日誌交付，您也需要下列其他許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowLogDeliveryActionsConsoleCWL",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:*"
            ]
        },
        {
            "Sid": "AllowLogDeliveryActionsConsoleS3",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::*"
            ]
        },
        {
            "Sid": "AllowLogDeliveryActionsConsoleFH",
            "Effect": "Allow",
            "Action": [
                "firehose:ListDeliveryStreams",
                "firehose:DescribeDeliveryStream"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

跨帳戶交付範例

在此範例中，涉及兩個帳戶。具有日誌產生資源的帳戶是帳戶 A，ID：*123456789012*，而具有日誌使用資源的帳戶是帳戶 B，ID：*111122223333*。

帳戶 A 想要使用 ARN arn：aws：bedrock：*us-east-1*：*123456789012*：knowledge-base/*kb-12345678* 從帳戶中的 Amazon Bedrock 知識庫傳遞日誌。

在此範例中，帳戶 A 需要下列許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowVendedLogDeliveryForKnowledgeBase",
            "Effect": "Allow",
            "Action": [
                "bedrock:AllowVendedLogDeliveryForResource"
            ],
            "Resource": "arn:aws:bedrock:us-east-1:123456789012:knowledge-
base/XXXXXXXXXX"
        },
        {
            "Sid": "CreateLogDeliveryPermissions",
            "Effect": "Allow",
            "Action": [
                "logs:PutDeliverySource",
                "logs:CreateDelivery"
            ],
            "Resource": [
            "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
    "arn:aws:logs:us-east-1:123456789012:delivery:*",
    "arn:aws:logs:us-east-1:444455556666:delivery-destination:*"
            ]
        }
    ]
}
```

建立交付來源

首先，帳戶 A 會建立交付來源及其基礎知識庫：

```
aws logs put-delivery-source --name my-delivery-source --log-type APPLICATION_LOGS --
resource-arn arn:aws:bedrock:region:AAAAAAAAAAAA:knowledge-base/XXXXXXXXXX
```

接下來，帳戶 B 必須使用下列其中一個流程建立交付目的地：

- 設定交付至 Amazon S3 儲存貯體
- 設定交付至 Firehose 串流

設定交付至 Amazon S3 儲存貯體

帳戶 B 想要使用 ARN arn：aws：s3：：amzn-s3-demo-bucket 接收其 S3 儲存貯體的日誌。在此範例中，帳戶 B 需要下列許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PutLogDestinationPermissions",
            "Effect": "Allow",
            "Action": [
                "logs:PutDeliveryDestination",
                "logs:PutDeliveryDestinationPolicy"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-
destination:*"
        }
    ]
}
```

儲存貯體在其儲存貯體政策中將需要下列許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogsDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/123456789012/
*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": [
                    "123456789012"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                    "arn:aws:logs:us-east-1:123456789012:delivery-source:my-
delivery-source"
                    ]
                }
            }
        }
    ]
}
```

如果儲存貯體使用 SSE-KMS 加密，請確保 AWS KMS 金鑰政策具有適當的許可。例如，如果 KMS 金鑰為 arn:aws:kms:*us-east-1*:*111122223333*:key/*1234abcd-12ab-34cd-56ef-1234567890ab*，請使用下列項目：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowLogsGenerateDataKey",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": [
                    "123456789012"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                    "arn:aws:logs:us-east-1:123456789012:delivery-source:my-
delivery-source"
                    ]
                }
            }
        }
    ]
}
```

然後，帳戶 B 可以使用 S3 儲存貯體做為目的地資源來建立交付目的地：

```
aws logs put-delivery-destination --name my-s3-delivery-destination --delivery-
destination-configuration "destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket"
```

接著，帳戶 B 在其新建立的交付目的地上建立交付目的地政策，這會授予帳戶 A 建立日誌交付的許可。將新增至新建立的交付目的地的政策如下：

JSON

```
    {
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateDelivery",
            "Effect": "Allow",
            "Principal": {
                "AWS": "123456789012"
            },
            "Action": [
                "logs:CreateDelivery"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-
destination:amzn-s3-demo-bucket"
        }
    ]
}
```

此政策將儲存在帳戶 B 的電腦中，做為destination-policy-s3.json連接此資源之用，帳戶 B 將執行下列命令：

```
aws logs put-delivery-destination-policy --delivery-destination-name my-s3-delivery-
destination --delivery-destination-policy file://destination-policy-s3.json
```

最後，帳戶 A 會建立交付，將帳戶 A 中的交付來源連結至帳戶 B 中的交付目的地。

```
aws logs create-delivery --delivery-source-name my-delivery-source --delivery-
destination-arn arn:aws:logs:region:BBBBBBBBBBBB:delivery-destination:my-s3-delivery-
destination
```

設定交付至 Firehose 串流

在此範例中，帳戶 B 想要將日誌接收到其 Firehose 串流。Firehose 串流具有下列 ARN，並設定為使用 DirectPut 交付串流類型：

arn:aws:firehose:*us-east-1*:*111122223333*:deliverystream/*log-delivery-stream*

在此範例中，帳戶 B 需要下列許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowFirehoseCreateSLR",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery"
        },
        {
            "Sid": "AllowFirehoseTagging",
            "Effect": "Allow",
            "Action": [
                "firehose:TagDeliveryStream"
            ],
            "Resource": "arn:aws:firehose:us-
east-1:111122223333:deliverystream/X"
        },
        {
            "Sid": "AllowFirehoseDeliveryDestination",
            "Effect": "Allow",
            "Action": [
                "logs:PutDeliveryDestination",
                "logs:PutDeliveryDestinationPolicy"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-
destination:*"
        }
    ]
}
```

Firehose 串流必須將標籤`LogDeliveryEnabled`設定為 `true`。

帳戶 B 接著會使用 Firehose 串流做為目的地資源來建立交付目的地：

```
aws logs put-delivery-destination --name my-fh-delivery-destination --delivery-
destination-configuration
 "destinationResourceArn=arn:aws:firehose:region:BBBBBBBBBBBB:deliverystream/X"
```

接著，帳戶 B 在其新建立的交付目的地上建立交付目的地政策，這會授予帳戶 A 建立日誌交付的許可。要新增至新建立的交付目的地的政策如下：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateDelivery",
            "Effect": "Allow",
            "Principal": {
            "AWS": "123456789012"
            },
            "Action": [
                "logs:CreateDelivery"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-
destination:amzn-s3-demo-bucket"
        }
    ]
}
```

此政策會儲存在帳戶 B 的電腦中，做為destination-policy-fh.json連接此資源之用，帳戶 B 會執行下列命令：

```
aws logs put-delivery-destination-policy --delivery-destination-name my-fh-delivery-
destination --delivery-destination-policy file://destination-policy-fh.json
```

最後，帳戶 A 會建立交付，將帳戶 A 中的交付來源連結至帳戶 B 中的交付目的地。

```
aws logs create-delivery --delivery-source-name my-delivery-source --delivery-
destination-arn arn:aws:logs:region:BBBBBBBBBBBB:delivery-destination:my-fh-delivery-
destination
```

預防跨服務混淆代理人

混淆代理人問題屬於安全性問題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在 中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了防止這種情況， AWS 提供工具，協助您保護所有 服務的資料，讓服務主體能夠存取您帳戶中的資源。

我們建議在資源政策中使用 aws:SourceArn、aws:SourceOrgID、 aws:SourceAccount和 aws:SourceOrgPaths全域條件內容索引鍵，以限制 CloudWatch Logs 為資源提供其他服務的許可。使用 aws:SourceArn，僅將一個資源與跨服務存取權相關聯。使用 aws:SourceAccount，讓該帳戶中的任何資源都與跨服務使用相關聯。使用 aws:SourceOrgID，允許組織內任何帳戶的任何資源與跨服務使用相關聯。使用 aws:SourceOrgPaths 將 AWS Organizations 路徑內帳戶中的任何資源與跨服務使用建立關聯。如需使用和了解路徑的詳細資訊，請參閱了解 AWS Organizations 實體路徑。

防範混淆代理人問題的最有效方法是使用 aws:SourceArn 全域條件內容索引鍵，以及資源的完整 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 aws:SourceArn 全域內容條件索引鍵搭配萬用字元 (*) 來表示 ARN 的未知部分。例如 arn:aws:*servicename*:*:*123456789012*:*。

如果 aws:SourceArn 值不包含帳戶 ID (例如 Amazon S3 儲存貯體 ARN)，您必須同時使用 aws:SourceAccount 和 aws:SourceArn 來限制許可。

若要大規模防範混淆代理人問題，請在資源型政策中使用 aws:SourceOrgID 或 aws:SourceOrgPaths 全域條件內容鍵和資源的組織 ID 或組織路徑。當您新增、移除或移動組織中的帳戶時，包含 aws:SourceOrgID 或 aws:SourceOrgPaths 鍵的政策將會自動包含正確的帳戶，您無需手動更新政策。

記載的政策用於授予對 CloudWatch Logs 的存取權，以將資料寫入 中的 Kinesis Data Streams 和 Firehose，步驟 1：建立目的地並步驟 2：建立目的地示範如何使用aws:SourceArn全域條件內容索引鍵來協助防止混淆代理人問題。

將日誌資料匯出至 Amazon S3

本章為您提供資訊，因此您可以將日誌資料從日誌群組匯出至 Amazon S3 儲存貯體，以進行自訂處理和分析，或載入至其他系統。您可以匯出至相同帳戶或不同帳戶中的 S3 儲存貯體。

您可以執行下列動作：

- 將日誌資料匯出至 SSE-KMS in AWS Key Management Service (AWS KMS) 加密的 S3 儲存貯體
- 將日誌資料匯出至啟用 S3 Object Lock 且具有保留期的 S3 儲存貯體。

我們建議您不要定期匯出至 Amazon S3，以持續封存您的日誌。對於該使用案例，我們建議您改用訂閱。如需訂閱的詳細資訊，請參閱 使用訂閱即時處理日誌資料。

若要開始匯出程序，您必須建立一個 S3 儲存貯體來存放匯出的日誌資料。您可以將匯出的檔案存放在 S3 儲存貯體，並定義 Amazon S3 生命週期規則以自動封存或刪除匯出的檔案。

您可以匯出至使用 AES-256 或 SSE-KMS 加密的 S3 儲存貯體。不支援匯出至使用 DSSE-KMS 加密的儲存貯體。

您可以從多個日誌群組或多個時間範圍，將日誌匯出至相同的 S3 儲存貯體。若要組織匯出的資料，請為每個匯出任務指定字首，該字首將用作所有匯出物件的 Amazon S3 金鑰字首。例如：`prod/app-logs/2026-01-03/` 或 `log-group-name/backup/`

> **Note**
>
> 無法保證將匯出檔案內的日誌資料塊依時間排序。您可以使用 Linux 公用程式對匯出的日誌欄位資料進行排序。例如，以下公用程式命令會對單一資料夾內所有 `.gz` 檔案中的事件進行排序。
>
> ```
> find . -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
> ```
>
> 以下公用程式命令對多個子資料夾內的 .gz 檔案進行排序。
>
> ```
> find ./*/ -type f -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
> ```
>
> 此外，您可以使用另一個 `stdout` 命令將排序後的輸出傳送到另一個檔案儲存。

日誌資料最長需要 12 個小時才能匯出。匯出任務會在 24 小時後逾時。如果匯出任務逾時，請縮短建立匯出任務時的時間範圍。

如需接近即時的日誌資料分析，請改為參閱使用 CloudWatch Logs Insights 分析日誌資料或使用訂閱即時處理日誌資料。

目錄

概念

您開始之前，請熟悉以下匯出概念：

日誌群組名稱

與匯出任務關聯的日誌群組名稱。此日誌群組中的日誌資料將匯出到指定的 S3 儲存貯體。

從 (時間戳記)

所需的時間戳記以從 1970 年 1 月 1 日 00:00:00 UTC 開始的毫秒數表示。日誌群組中在此時間或之後擷取的所有日誌事件都會匯出。

至 (時間戳記)

所需的時間戳記以從 1970 年 1 月 1 日 00:00:00 UTC 開始的毫秒數表示。此日誌群組中在此時間之前擷取的所有日誌事件都會匯出。

目的地儲存貯體

與匯出任務關聯的 S3 儲存貯體名稱。此儲存貯體用於從指定的日誌群組匯出日誌資料。

目的地前綴

選用的屬性，作為所有匯出物件的 Amazon S3 金鑰前綴。這有助於在您的儲存貯體建立類似資料夾的整理方式。

使用主控台將日誌資料匯出至 Amazon S3

在以下範例中，您會使用 Amazon CloudWatch 主控台，從名為 `my-log-group` 的 Amazon CloudWatch Logs 日誌群組將所有資料匯出至名為 `amzn-s3-demo-bucket` 的 Amazon S3 儲存貯體。

支援將日誌資料匯出至由 SSE-KMS 加密的 S3 儲存貯體。不支援匯出至使用 DSSE-KMS 加密的儲存貯體。

如何設定匯出作業的詳細方法，取決於您要存放匯出資料的 Amazon S3 儲存貯體是否與要匯出的日誌位於同一帳戶。

主題

相同帳戶匯出（主控台）

如果 Amazon S3 儲存貯體與要匯出的日誌位於同一帳戶，請參閱本區段的說明。

主題

建立 Amazon S3 儲存貯體（主控台）

我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，您可以跳到步驟 2。

> **Note**
>
> Amazon S3 儲存貯體與您要匯出的日誌資料，必須位於相同的區域。CloudWatch Logs 不支援將資料匯出至不同區域中的 Amazon S3 儲存貯體。

建立 Amazon S3 儲存貯體

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 如有必要請變更區域。從導覽列，選擇您的 CloudWatch Logs 所在的區域。
3. 選擇 Create Bucket (建立儲存貯體)。
4. 針對 Bucket Name (儲存貯體名稱)，輸入儲存貯體的名稱。
5. 針對 Region (區域)，選取您的 CloudWatch Logs 資料所在的區域。
6. 選擇建立。

設定存取許可 （主控台）

若要建立匯出任務，您需要使用 IAM `AmazonS3ReadOnlyAccess` 角色和下列許可登入：

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

設定 Amazon S3 儲存貯體的許可 （主控台）

根據預設，所有 Amazon S3 儲存貯體和物件皆為私有。只有建立儲存貯 AWS 帳戶 體的資源擁有者，才能存取儲存貯體及其包含的任何物件。不過，資源擁有者可藉由編寫存取政策，選擇將存取許可授予其他資源或使用者。

當您設定政策時，我們建議您包含隨機產生的字串做為儲存貯體的前綴，如此一來，只有適用的日誌串流才會匯出到儲存貯體。

> **Important**
>
> 為了讓匯出至 Amazon S3 儲存貯體更安全，我們現在會要求您指定允許將日誌資料匯出至 S3 儲存貯體的來源帳戶清單。
> 在下列範例中，`aws:SourceAccount`金鑰中的帳戶 IDs 清單會是使用者可以將日誌資料匯出至 Amazon S3 儲存貯體的帳戶。`aws:SourceArn` 金鑰會是正在採取行動的資源。您可以將其限制為具體的日誌群組，或使用萬用字元，如本範例所示。
> 我們建議您也包含建立 S3 儲存貯體之帳戶的帳戶 ID，以允許在同一帳戶內匯出。

設定 Amazon S3 儲存貯體的許可

1. 在 Amazon S3 主控台中，選擇您建立的儲存貯體。
2. 選擇 Permissions (許可)、Bucket policy (儲存貯體政策)。
3. 在 Bucket Policy Editor (儲存貯體政策編輯器) 中，輸入下列政策。將 `amzn-s3-demo-bucket` 變更為 Amazon S3 儲存貯體的名稱。務必為主體指定正確的區域端點，例如 `us-west-1`。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
      {
          "Sid": "AllowCloudWatchLogsGetBucketAcl",
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
          "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
          "Condition": {
```

```
            "StringEquals": {
                "aws:SourceAccount": [
                    "123456789012",
                    "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:123456789012:log-group:*",
                        "arn:aws:logs:us-east-1:111122223333:log-group:*"
                    ]
            }
        }
    },
    {
        "Sid": "AllowCloudWatchLogsPutObject",
        "Action": "s3:PutObject",
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
        "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                    "123456789012",
                    "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:123456789012:log-group:*",
                        "arn:aws:logs:us-east-1:111122223333:log-group:*"
                    ]
            }
        }
    }
]
}
```

4. 選擇 Save (儲存)，將您剛才新增的政策設定為您儲存貯體上的存取政策。此政策可讓 CloudWatch Logs 將日誌資料匯出至您的 Amazon S3 儲存貯體。儲存貯體擁有者擁有所有匯出物件的完整許可。

> **Warning**
>
> 如果現有的儲存貯體已連接一個或多個政策，請新增陳述式讓 CloudWatch Logs 存取這個或這些政策。我們建議您評估所產生的一組許可，以確保它們適用於將存取儲存貯體的使用者。

（選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體（主控台）

只有在您匯出至使用伺服器端加密的 Amazon S3 儲存貯體時，才需要此步驟 AWS KMS keys。這種加密稱為 SSE-KMS。

匯出至使用 SSE-KMS 加密的儲存貯體

1. 在 https：//https://console.aws.amazon.com/kms 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在左側導覽列中，選擇 Customer managed keys (客戶受管金鑰)。

   選擇 Create Key (建立金鑰)。
4. 針對 Key type (金鑰類型)，請選擇 Symmetric (對稱)。
5. 在 Key usage (金鑰用途) 中，選擇 Encrypt and decrypt (加密與解密)，然後選擇 Next (下一步)。
6. 在 Add labels (加入標示) 下，輸入金鑰的別名，並選擇是否新增說明或標籤。然後選擇下一步。
7. 在 Key administrators (金鑰管理員) 下，選取可管理此金鑰的人員，然後選擇 Next (下一步)。
8. 在 Define key usage permissions (定義金鑰用途許可) 下，不進行變更，然後選擇 Next (下一步)。
9. 檢閱設定，然後選擇 Finish (完成)。
10. 返回 Customer managed keys (客戶受管金鑰) 頁面，選擇您剛建立的金鑰名稱。
11. 選擇 Key policy (金鑰政策) 索引標籤，並選擇 Switch to policy view (切換至政策檢視)。
12. 在 Key policy (金鑰政策) 區段中，選擇 Edit (編輯)。
13. 將下列陳述式新增至金鑰政策陳述式清單。執行時，請將 *Region* 替換為日誌區域，並以所擁有 KMS 金鑰的帳戶 ARN 替換 *account-ARN*。

    JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow CWL Service Principal usage",
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.Region.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "account-ARN"
            },
            "Action": [
                "kms:GetKeyPolicy*",
                "kms:PutKeyPolicy*",
                "kms:DescribeKey*",
                "kms:CreateAlias*",
                "kms:ScheduleKeyDeletion*",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

14. 選擇儲存變更。
15. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
16. 尋找您在 建立 S3 儲存貯體 (CLI) 中建立的儲存貯體，選擇儲存貯體名稱。
17. 選擇屬性索引標籤。在 Default encryption (預設加密) 下，選擇 Edit (編輯)。
18. 在 Server-side Encryption (伺服器端加密) 下，選擇 Enable (啟用)。
19. 在 Encryption type (加密類型) 下，選擇 AWS Key Management Service key (SSE-KMS) (金鑰 (SSE-KMS))。
20. 選擇從 AWS KMS 金鑰中選擇，然後尋找您建立的金鑰。

21. 在 Bucket Key (儲存貯體金鑰) 下，選擇 Enable (啟用)。
22. 選擇儲存變更。

建立匯出任務 （主控台）

在此程序中，您會建立匯出任務，以從日誌群組匯出日誌。

使用 CloudWatch 主控台將資料匯出至 Amazon S3

1. 如 設定存取許可 （主控台） 中所示，以足夠的許可登入。
2. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
3. 在導覽窗格中，選擇 Log groups (日誌群組)。
4. 在 Log Groups (日誌群組) 畫面上，選擇日誌群組的名稱。
5. 選擇 Actions (動作)、Export data to Amazon S3 (匯出資料至 Amazon S3)。
6. 在 Export data to Amazon S3 (匯出資料至 Amazon S3) 畫面的 Define data to export (定義資料匯出) 下方，使用 From (從) 和 To (至) 設定所要匯出資料的時間範圍。
7. 如果您的日誌群組有多個日誌串流，您可以提供日誌串流前綴，將日誌群組資料限制於特定串流。選擇 Advanced (進階)，然後針對 Stream prefix (串流前綴)，輸入日誌串流前綴。
8. 在 Choose S3 bucket (選擇 S3 儲存貯體) 下，選擇與 S3 儲存貯體關聯的帳戶。
9. 為 S3 bucket name (S3 儲存貯體名稱) 選擇一個 &S3; 儲存貯體。
10. 針對 S3 Bucket prefix (S3 儲存貯體前綴)，輸入您在儲存貯體政策中指定的隨機產生字串。
11. 選擇 Export (匯出) 將您的日誌資料匯出至 Amazon S3。
12. 若要檢視您匯出至 Amazon S3 的日誌資料的狀態，請選擇 Actions (動作，然後選擇 View all exports to Amazon S3 (檢視所有匯出至 Amazon S3 的項目)。

跨帳戶匯出 （主控台）

如果 Amazon S3 儲存貯體與要匯出的日誌位於不同帳戶，請參閱本區段的說明。

主題

- 建立跨帳戶匯出的 Amazon S3 儲存貯體 （主控台）
- 設定跨帳戶匯出的存取許可 （主控台）
- 在 S3 儲存貯體上設定跨帳戶匯出的許可 （主控台）

- （選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體以進行跨帳戶匯出（主控台）
- 建立跨帳戶匯出的匯出任務（主控台）

建立跨帳戶匯出的 Amazon S3 儲存貯體（主控台）

我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，可以略過此程序。

> **Note**
>
> Amazon S3 儲存貯體與您要匯出的日誌資料，必須位於相同的區域。CloudWatch Logs 不支援將資料匯出至不同區域中的 Amazon S3 儲存貯體。

建立 Amazon S3 儲存貯體

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 如有必要請變更區域。從導覽列，選擇您的 CloudWatch Logs 所在的區域。
3. 選擇 Create Bucket (建立儲存貯體)。
4. 針對 Bucket Name (儲存貯體名稱)，輸入儲存貯體的名稱。
5. 針對 Region (區域)，選取您的 CloudWatch Logs 資料所在的區域。
6. 選擇建立。

設定跨帳戶匯出的存取許可（主控台）

首先，您必須建立新的 IAM 政策，讓 CloudWatch Logs 在目的地帳戶中擁有目的地 Amazon S3 儲存貯體`s3:PutObject`的動作。

除了 `s3:PutObject`動作之外，政策中包含的其他動作取決於目的地儲存貯體是否使用 AWS KMS 加密，或使用 S3 物件擁有權設定啟用 ACLs。

- 如果使用 KMS 加密，請新增金鑰資源的 `kms:GenerateDataKey`和 `kms:Decrypt`動作
- 如果在儲存貯體上啟用 ACLs請新增儲存貯體資源`s3:PutObjectAcl`的動作

在下列政策中`amzn-s3-demo-bucket`，將 變更為目的地 S3 儲存貯體的名稱。

若要建立 IAM 政策將日誌匯出至 Amazon S3 儲存貯體

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策。
4. 在政策編輯器區段中，選擇 JSON。
5. 如果目的地儲存貯體不使用 AWS KMS 加密，請將下列政策貼入編輯器。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

如果目的地儲存貯體確實使用 AWS KMS 加密，請將下列政策貼入編輯器。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
```

```
      "Resource": "arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}
```

如果已在目的地儲存貯體上啟用 ACLs，請在上述政策中將 s3：PutObjectAcl 新增至 s3：PutObject 動作區塊。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:PutObject",
               "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

6. 選擇下一步。
7. 輸入政策名稱。您會使用此名稱為您的 IAM 角色附加政策。
8. 選擇建立政策，儲存新政策。

若要建立匯出任務，您必須使用已連接 AmazonS3ReadOnlyAccess受管政策的 IAM 角色、上述建立的 IAM 政策，以及具有下列許可的 IAM 角色登入：

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

在 S3 儲存貯體上設定跨帳戶匯出的許可 （主控台）

依據預設，所有 S3 儲存貯體與物件皆為私有。只有建立儲存貯 AWS 帳戶 體的資源擁有者，才能存取儲存貯體及其包含的任何物件。不過，資源擁有者可藉由編寫存取政策，選擇將存取許可授予其他資源或使用者。

當您設定政策時，我們建議您包含隨機產生的字串做為儲存貯體的前綴，如此一來，只有適用的日誌串流才會匯出到儲存貯體。

> **Important**
>
> 為了使匯出至 S3 儲存貯體更加安全，現在要求您指定允許將日誌資料匯出至 S3 儲存貯體的來源帳戶清單。
> 在下列範例中，`aws:SourceAccount`金鑰中的帳戶 IDs 清單會是使用者可以將日誌資料匯出至 S3 儲存貯體的帳戶。`aws:SourceArn` 金鑰會是正在採取行動的資源。您可以將其限制為具體的日誌群組，或使用萬用字元，如本範例所示。
> 我們建議您也包含建立 S3 儲存貯體之帳戶的帳戶 ID，以允許在同一帳戶內匯出。

設定 Amazon S3 儲存貯體的許可

1. 在 Amazon S3 主控台中，選擇您建立的儲存貯體。
2. 選擇 Permissions (許可)、Bucket policy (儲存貯體政策)。
3. 在 Bucket Policy Editor (儲存貯體政策編輯器) 中，輸入下列政策。將 `amzn-s3-demo-bucket` 變更為 Amazon S3 儲存貯體的名稱。務必為主體指定正確的區域端點，例如 `us-east-1`。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
          "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "aws:SourceAccount": [
                    "123456789012",
                    "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:123456789012:log-group:*",
                        "arn:aws:logs:us-east-1:111122223333:log-group:*"
                    ]
            }
          }
      },
      {
          "Action": "s3:PutObject",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
          "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                    "123456789012",
                    "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:us-east-1:123456789012:log-group:*",
```

```
                    "arn:aws:logs:us-east-1:111122223333:log-group:*"
                ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Principal": {
          "AWS": "arn:aws:iam::111122223333:role/role_name"
        },
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "Condition": {
          "StringEquals": {
              "s3:x-amz-acl": "bucket-owner-full-control"
          }
        }
    }
  ]
}
```

4. 選擇 Save (儲存)，將您剛才新增的政策設定為您儲存貯體上的存取政策。此政策可讓 CloudWatch Logs 將日誌資料匯出至您的 S3 儲存貯體。儲存貯體擁有者擁有所有匯出物件的完整許可。

> **Warning**
>
> 如果現有的儲存貯體已連接一個或多個政策，請新增陳述式讓 CloudWatch Logs 存取這個或這些政策。我們建議您評估所產生的一組許可，以確保它們適用於將存取儲存貯體的使用者。

（選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體以進行跨帳戶匯出（主控台）

只有在您匯出至使用伺服器端加密的 S3 儲存貯體時，才需要此程序 AWS KMS keys。這種加密稱為 SSE-KMS。

匯出至使用 SSE-KMS 加密的儲存貯體

1. 在 https：//https://console.aws.amazon.com/kms 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在左側導覽列中，選擇 Customer managed keys (客戶受管金鑰)。

   選擇 Create Key (建立金鑰)。
4. 針對 Key type (金鑰類型)，請選擇 Symmetric (對稱)。
5. 在 Key usage (金鑰用途) 中，選擇 Encrypt and decrypt (加密與解密)，然後選擇 Next (下一步)。
6. 在 Add labels (加入標示) 下，輸入金鑰的別名，並選擇是否新增說明或標籤。然後選擇下一步。
7. 在 Key administrators (金鑰管理員) 下，選取可管理此金鑰的人員，然後選擇 Next (下一步)。
8. 在 Define key usage permissions (定義金鑰用途許可) 下，不進行變更，然後選擇 Next (下一步)。
9. 檢閱設定，然後選擇 Finish (完成)。
10. 返回 Customer managed keys (客戶受管金鑰) 頁面，選擇您剛建立的金鑰名稱。
11. 選擇 Key policy (金鑰政策) 索引標籤，並選擇 Switch to policy view (切換至政策檢視)。
12. 在 Key policy (金鑰政策) 區段中，選擇 Edit (編輯)。
13. 將下列陳述式新增至金鑰政策陳述式清單。當您這麼做時，請將 *us-east-1* 取代為日誌的區域，將 *account-ARN* 取代為擁有 KMS 金鑰的帳戶 ARN，將 *123456789012* 取代為擁有 KMS 金鑰的帳號，將*key_id* 取代為用於建立匯出任務的角色。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow CWL Service Principal usage",
            "Effect": "Allow",
            "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "account-ARN"
            },
            "Action": [
                "kms:GetKeyPolicy*",
                "kms:PutKeyPolicy*",
                "kms:DescribeKey*",
                "kms:CreateAlias*",
                "kms:ScheduleKeyDeletion*",
                "kms:Decrypt"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Enable IAM Role Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/role_name"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-1:123456789012:key/key-id"
        }
    ]
}
```

14. 選擇儲存變更。
15. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
16. 尋找您在 建立 S3 儲存貯體 (CLI) 中建立的儲存貯體，選擇儲存貯體名稱。
17. 選擇屬性索引標籤。在 Default encryption (預設加密) 下，選擇 Edit (編輯)。
18. 在 Server-side Encryption (伺服器端加密) 下，選擇 Enable (啟用)。
19. 在 Encryption type (加密類型) 下，選擇 AWS Key Management Service key (SSE-KMS) (金鑰 (SSE-KMS))。
20. 選擇從 AWS KMS 金鑰中選擇，然後尋找您建立的金鑰。
21. 在 Bucket Key (儲存貯體金鑰) 下，選擇 Enable (啟用)。

22. 選擇儲存變更。

建立跨帳戶匯出的匯出任務（主控台）

在此程序中，您會建立匯出任務，以從日誌群組匯出日誌。

使用 CloudWatch 主控台將資料匯出至 Amazon S3

1. 如 設定存取許可（主控台） 中所示，以足夠的許可登入。
2. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
3. 在導覽窗格中，選擇 Log groups (日誌群組)。
4. 在 Log Groups (日誌群組) 畫面上，選擇日誌群組的名稱。
5. 選擇 Actions (動作)、Export data to Amazon S3 (匯出資料至 Amazon S3)。
6. 在 Export data to Amazon S3 (匯出資料至 Amazon S3) 畫面的 Define data to export (定義資料匯出) 下方，使用 From (從) 和 To (至) 設定所要匯出資料的時間範圍。
7. 如果您的日誌群組有多個日誌串流，您可以提供日誌串流前綴，將日誌群組資料限制於特定串流。選擇 Advanced (進階)，然後針對 Stream prefix (串流前綴)，輸入日誌串流前綴。
8. 在 Choose S3 bucket (選擇 S3 儲存貯體) 下，選擇與 S3 儲存貯體關聯的帳戶。
9. 在 S3 bucket name (S3 儲存貯體名稱) 中，選擇一個 S3 儲存貯體。
10. 針對 S3 Bucket prefix (S3 儲存貯體前綴)，輸入您在儲存貯體政策中指定的隨機產生字串。
11. 選擇 Export (匯出) 將您的日誌資料匯出至 Amazon S3。
12. 若要檢視您匯出至 Amazon S3 的日誌資料的狀態，請選擇 Actions (動作，然後選擇 View all exports to Amazon S3 (檢視所有匯出至 Amazon S3 的項目)。

使用 將日誌資料匯出至 Amazon S3 AWS CLI

在以下範例中，您將使用匯出任務，從名為 `my-log-group` 的 CloudWatch Logs 日誌群組，將所有資料匯出至名為 `amzn-s3-demo-bucket` 的 Amazon S3 儲存貯體。此範例假設您已建立一個名為 `my-log-group` 的日誌群組。

AWS KMS 支援將日誌資料匯出至 加密的 S3 儲存貯體。不支援匯出至使用 DSSE-KMS 加密的儲存貯體。

如何設定匯出作業的詳細方法，取決於您要存放匯出資料的 Amazon S3 儲存貯體是否與要匯出的日誌位於同一帳戶。

主題

相同帳戶匯出 (CLI)

如果 Amazon S3 儲存貯體與要匯出的日誌位於同一帳戶，請參閱本區段的說明。

主題

建立 S3 儲存貯體 (CLI)

我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，可以略過此程序。

> **Note**
>
> S3 儲存貯體必須與要匯出的日誌資料位於相同的區域。CloudWatch Logs 不支援將資料匯出至不同區域中的 S3 儲存貯體。

使用 建立 S3 儲存貯體 AWS CLI

在命令提示中執行以下 create-bucket 命令，其中的 `LocationConstraint` 是您要匯出日誌資料的區域。

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-2
```

以下為範例輸出。

```
{
```

```
    "Location": "/amzn-s3-demo-bucket"
}
```

設定存取許可 (CLI)

若要稍後建立匯出任務，您需要使用 IAM AmazonS3ReadOnlyAccess 角色和下列許可登入：

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

設定 S3 儲存貯體的許可 (CLI)

依據預設，所有 S3 儲存貯體與物件皆為私有。只有資源擁有者、建立儲存貯體的帳戶，才能存取儲存貯體及其包含的任何物件。不過，資源擁有者可藉由編寫存取政策，選擇將存取許可授予其他資源或使用者。

> **Important**
>
> 為了使匯出至 S3 儲存貯體更加安全，現在要求您指定允許將日誌資料匯出至 S3 儲存貯體的來源帳戶清單。

在下列範例中，aws:SourceAccount金鑰中的帳戶 IDs 清單會是使用者可以將日誌資料匯出至 S3 儲存貯體的帳戶。aws:SourceArn 金鑰會是正在採取行動的資源。您可以將其限制為具體的日誌群組，或使用萬用字元，如本範例所示。
我們建議您也包含建立 S3 儲存貯體之帳戶的帳戶 ID，以允許在同一帳戶內匯出。

設定 S3 儲存貯體的許可

1. 建立名為 policy.json 的檔案，然後新增以下存取政策，將 amzn-s3-demo-bucket 變更為您的 S3 儲存貯體名稱，然後將 Principal 變更為您匯出日誌資料的區域端點，例如us-east-1 。請使用文字編輯器來建立此政策檔案。請勿使用 IAM 主控台。

   JSON

```
{
"Version":"2012-10-17",
"Statement": [
  {
      "Sid": "AllowGetBucketAcl",
      "Action": "s3:GetBucketAcl",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
            "aws:SourceAccount": [
                "123456789012",
                "111122223333"
            ]
        },
        "ArnLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:us-east-1:123456789012:log-group:*",
                    "arn:aws:logs:us-east-1:111122223333:log-group:*"
                ]
        }
      }
  },
  {
      "Sid": "AllowPutObject",
```

```
            "Action": "s3:PutObject",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
            "Condition": {
              "StringEquals": {
                  "s3:x-amz-acl": "bucket-owner-full-control",
                  "aws:SourceAccount": [
                      "123456789012",
                      "111122223333"
                  ]
              },
              "ArnLike": {
                      "aws:SourceArn": [
                          "arn:aws:logs:us-east-1:123456789012:log-group:*",
                          "arn:aws:logs:us-east-1:111122223333:log-group:*"
                      ]
              }
            }
        }
    ]
}
```

2. 使用 put-bucket-policy 命令，將您剛才新增的政策設定為您儲存貯體上的存取政策。此政策可讓 CloudWatch Logs 將日誌資料匯出至您的 S3 儲存貯體。儲存貯體擁有者將擁有所有匯出物件的完整許可。

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://
policy.json
```

> **Warning**
>
> 如果現有的儲存貯體已連接一個或多個政策，請新增陳述式讓 CloudWatch Logs 存取這個或這些政策。我們建議您評估所產生的一組許可，以確保它們適用於將存取儲存貯體的使用者。

（選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體 (CLI)

只有在您匯出至使用伺服器端加密的 S3 儲存貯體時，才需要此程序 AWS KMS keys。這種加密稱為 SSE-KMS。

匯出至使用 SSE-KMS 加密的儲存貯體

1. 使用文字編輯器建立名為 key_policy.json 的檔案，並新增下列存取政策。新增政策時，進行下列變更：

   - 將 *Region* 替換成您的日誌區域。
   - 將 *account-ARN* 替換成所擁有 KMS 金鑰的帳戶 ARN。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow CWL Service Principal usage",
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.Region.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "account-ARN"
            },
            "Action": [
                "kms:GetKeyPolicy*",
                "kms:PutKeyPolicy*",
                "kms:DescribeKey*",
                "kms:CreateAlias*",
                "kms:ScheduleKeyDeletion*",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

2. 輸入以下命令：

```
aws kms create-key --policy file://key_policy.json
```

以下為此命令的範例輸出：

```
{
    "KeyMetadata": {
        "AWSAccountId": "account_id",
        "KeyId": "key_id",
        "Arn": "arn:aws:kms:us-east-2:account-ARN:key/key_id",
        "CreationDate": "time",
        "Enabled": true,
        "Description": "",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegion": false
    }
```

3. 使用文字編輯器建立名為 bucketencryption.json 的檔案，包含下列內容。

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

4. 輸入下列命令，以您要匯出日誌的儲存貯體名稱取代 *amzn-s3-demo-bucket*。

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-
encryption-configuration file://bucketencryption.json
```

如果命令沒有傳回錯誤，則表示該流程成功。

建立匯出任務 (CLI)

使用以下命令建立匯出任務。在您建立匯出任務後，此任務可能需花費幾秒到幾小時的時間，視匯出資料的大小而定。

使用 將資料匯出至 Amazon S3 AWS CLI

1. 如 設定存取許可 (CLI) 中所示，以足夠的許可登入。
2. 在命令提示中，使用下列 create-export-task 命令以建立匯出任務。

```
aws logs create-export-task --profile CWLExportUser --task-name "my-log-
group-09-10-2015" --log-group-name "my-log-group" --from 1441490400000 --
to 1441494000000 --destination "amzn-s3-demo-bucket" --destination-prefix "export-
task-output"
```

以下為範例輸出。

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

跨帳戶匯出 (CLI)

如果 Amazon S3 儲存貯體與要匯出的日誌位於不同帳戶，請參閱本區段的說明。

主題

- 建立 S3 儲存貯體以進行跨帳戶匯出 (CLI)
- 設定跨帳戶匯出的存取許可 (CLI)
- 設定 S3 儲存貯體的跨帳戶匯出許可 (CLI)
- （選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體以進行跨帳戶匯出 (CLI)

- 建立跨帳戶匯出的匯出任務 (CLI)

建立 S3 儲存貯體以進行跨帳戶匯出 (CLI)

我們建議您使用專為 CloudWatch Logs 建立的儲存貯體。不過，如果您想要使用現有的儲存貯體，您可以跳到步驟 2。

> **Note**
>
> S3 儲存貯體必須與要匯出的日誌資料位於相同的區域。CloudWatch Logs 不支援將資料匯出至不同區域中的 S3 儲存貯體。

使用 建立 S3 儲存貯體 AWS CLI

在命令提示中執行以下 create-bucket 命令，其中的 `LocationConstraint` 是您要匯出日誌資料的區域。

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-2
```

以下為範例輸出。

```
{
    "Location": "/amzn-s3-demo-bucket"
}
```

設定跨帳戶匯出的存取許可 (CLI)

首先，您必須建立新的 IAM 政策，讓 CloudWatch Logs 在目的地帳戶中擁有目的地 Amazon S3 儲存貯體`s3:PutObject`的動作。

除了 `s3:PutObject`動作之外，政策中包含的其他動作取決於目的地儲存貯體是否使用 AWS KMS 加密，或使用 S3 物件擁有權設定啟用 ACLs。

- 如果使用 KMS 加密，請新增金鑰資源的 `kms:GenerateDataKey`和 `kms:Decrypt`動作
- 如果在儲存貯體上啟用 ACLs請新增儲存貯體資源`s3:PutObjectAcl`的動作

在下列政策中`amzn-s3-demo-bucket`，將 變更為目的地 S3 儲存貯體的名稱。

您建立的政策取決於目的地儲存貯體是否使用 AWS KMS 加密。如果不使用 AWS KMS 加密，請使用下列內容建立政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

如果目的地儲存貯體使用 AWS KMS 加密，請使用下列內容建立政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

如果已在目的地儲存貯體上啟用 ACLs，請在上述政策中將 s3：PutObjectAcl 新增至 s3：PutObject 動作區塊。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:PutObject",
               "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

若要建立匯出任務，您必須使用已連接 AmazonS3ReadOnlyAccess受管政策的 IAM 角色、上述建立的 IAM 政策，以及具有下列許可的 IAM 角色登入：

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。

- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。

- IAM 使用者：

- 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
- (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

設定 S3 儲存貯體的跨帳戶匯出許可 (CLI)

依據預設，所有 S3 儲存貯體與物件皆為私有。只有資源擁有者、建立儲存貯體的帳戶，才能存取儲存貯體及其包含的任何物件。不過，資源擁有者可藉由編寫存取政策，選擇將存取許可授予其他資源或使用者。

> **Important**
>
> 為了使匯出至 S3 儲存貯體更加安全，現在要求您指定允許將日誌資料匯出至 S3 儲存貯體的來源帳戶清單。
> 在下列範例中，aws:SourceAccount金鑰中的帳戶 IDs 清單會是使用者可以將日誌資料匯出至 S3 儲存貯體的帳戶。aws:SourceArn 金鑰會是正在採取行動的資源。您可以將其限制為具體的日誌群組，或使用萬用字元，如本範例所示。
> 我們建議您也包含建立 S3 儲存貯體之帳戶的帳戶 ID，以允許在同一帳戶內匯出。

設定 S3 儲存貯體的許可

1. 建立名為 的檔案，policy.json並將下列存取政策變更為目的地 S3 儲存貯體amzn-s3-demo-bucket的名稱，Principal並新增至您要匯出日誌資料的 區域的端點，例如 us-west-1。請使用文字編輯器來建立此政策檔案。請勿使用 IAM 主控台。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
          "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
          "Condition": {
```

```
            "StringEquals": {
                "aws:SourceAccount": [
                "123456789012",
                "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                    "arn:aws:logs:us-east-1:123456789012:log-group:*",
                    "arn:aws:logs:us-east-1:111122223333:log-group:*"
                     ]
            }
        }
    },
    {
        "Action": "s3:PutObject" ,
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
        "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                "123456789012",
                "111122223333"
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                    "arn:aws:logs:us-east-1:123456789012:log-group:*",
                    "arn:aws:logs:us-east-1:111122223333:log-group:*"
                    ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role_name"
        },
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::>amzn-s3-demo-bucket/*",
        "Condition": {
            "StringEquals": {
```

```
                "s3:x-amz-acl": "bucket-owner-full-control"
            }
        }
    }
  ]
}
```

2. 使用 put-bucket-policy 命令，將您剛才新增的政策設定為您儲存貯體上的存取政策。此政策可讓 CloudWatch Logs 將日誌資料匯出至您的 S3 儲存貯體。儲存貯體擁有者將擁有所有匯出物件的完整許可。

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://
policy.json
```

> **Warning**
>
> 如果現有的儲存貯體已連接一個或多個政策，請新增陳述式讓 CloudWatch Logs 存取這個或這些政策。我們建議您評估所產生的一組許可，以確保它們適用於將存取儲存貯體的使用者。

（選用）匯出至使用 SSE-KMS 加密的目的地 Amazon S3 儲存貯體以進行跨帳戶匯出 (CLI)

只有在您匯出至使用伺服器端加密的 S3 儲存貯體時，才需要此程序 AWS KMS keys。這種加密稱為 SSE-KMS。

匯出至使用 SSE-KMS 加密的儲存貯體

1. 使用文字編輯器建立名為 `key_policy.json` 的檔案，並新增下列存取政策。新增政策時，進行下列變更：

   - 將 *`us-east-1`* 取代為日誌的區域。
   - 將 *`account-ARN`* 替換成所擁有 KMS 金鑰的帳戶 ARN。
   - 將 *`123456789012`* 取代為擁有 KMS 金鑰的帳號。
   - *`key_id`* 與 kms-key ID。
   - *`role_name`* 與用於建立匯出任務的角色。

JSON

```
{
"Version":"2012-10-17",
"Statement": [
    {
        "Sid": "AllowCWLServicePrincipalUsage",
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
        },
        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "*"
    },
    {
        "Sid": "EnableIAMUserPermissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "account-ARN"
        },
        "Action": [
            "kms:GetKeyPolicy*",
            "kms:PutKeyPolicy*",
            "kms:DescribeKey*",
            "kms:CreateAlias*",
            "kms:ScheduleKeyDeletion*",
            "kms:Decrypt"
        ],
        "Resource": "*"
    },
    {
        "Sid": "EnableIAMRolePermissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/role_name"
        },
        "Action": [
```

```
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

2. 輸入以下命令：

```
aws kms create-key --policy file://key_policy.json
```

以下為此命令的範例輸出：

```
{
    "KeyMetadata": {
        "AWSAccountId": "account_id",
        "KeyId": "key_id",
        "Arn": "arn:aws:kms:us-east-1:123456789012:key/key_id",
        "CreationDate": "time",
        "Enabled": true,
        "Description": "",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegion": false
    }
```

3. 使用文字編輯器建立名為 bucketencryption.json 的檔案，包含下列內容。

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
```

```
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

4. 輸入下列命令，以您要匯出日誌的儲存貯體名稱取代 *amzn-s3-demo-bucket*。

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-
encryption-configuration file://bucketencryption.json
```

如果命令沒有傳回錯誤，則表示該流程成功。

建立跨帳戶匯出的匯出任務 (CLI)

使用以下命令建立匯出任務。在您建立匯出任務後，此任務可能需花費幾秒到幾小時的時間，視匯出資料的大小而定。

使用 將資料匯出至 Amazon S3 AWS CLI

1. 如 設定存取許可 (CLI) 中所示，以足夠的許可登入。
2. 在命令提示中，使用下列 create-export-task 命令以建立匯出任務。

```
aws logs create-export-task --profile CWLExportUser --task-name "my-log-
group-09-10-2015" --log-group-name "my-log-group" --from 1441490400000 --
to 1441494000000 --destination "amzn-s3-demo-bucket" --destination-prefix "export-
task-output"
```

以下為範例輸出。

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

描述匯出任務 (CLI)

在您建立匯出任務之後，您可以取得任務的目前狀態。

使用 描述匯出任務 AWS CLI

在命令提示中，使用以下 describe-export-tasks 命令。

```
aws logs --profile CWLExportUser describe-export-tasks --task-id
 "cda45419-90ea-4db5-9833-aade86253e66"
```

以下為範例輸出。

```
{
   "exportTasks": [
   {
      "destination": "my-exported-logs",
      "destinationPrefix": "export-task-output",
      "executionInfo": {
         "creationTime": 1441495400000
      },
      "from": 1441490400000,
      "logGroupName": "my-log-group",
      "status": {
         "code": "RUNNING",
         "message": "Started Successfully"
      },
      "taskId": "cda45419-90ea-4db5-9833-aade86253e66",
      "taskName": "my-log-group-09-10-2015",
      "tTo": 1441494000000
   }]
}
```

您有三種使用 describe-export-tasks 命令的方法：

- 無任何篩選條件 – 以建立順序相反的順序列出您所有的匯出任務。
- 任務 ID 篩選 – 列出指定 ID 的匯出任務 (如果有的話)。
- 任務狀態篩選 – 列出指定狀態的匯出任務。

例如，使用以下命令來篩選 FAILED 狀態。

```
aws logs --profile CWLExportUser describe-export-tasks --status-code "FAILED"
```

以下為範例輸出。

```
{
```

```
   "exportTasks": [
   {
      "destination": "amzn-s3-demo-bucket",
      "destinationPrefix": "export-task-output",
      "executionInfo": {
         "completionTime": 1441498600000
         "creationTime": 1441495400000
      },
      "from": 1441490400000,
      "logGroupName": "my-log-group",
      "status": {
         "code": "FAILED",
         "message": "FAILED"
      },
      "taskId": "cda45419-90ea-4db5-9833-aade86253e66",
      "taskName": "my-log-group-09-10-2015",
      "to": 1441494000000
   }]
}
```

取消匯出任務 (CLI)

如果匯出任務處於 PENDING 或 RUNNING 狀態，可以取消匯出任務。

使用 取消匯出任務 AWS CLI

在命令提示字元中，使用下列 cancel-export-task 命令：

```
aws logs --profile CWLExportUser cancel-export-task --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

您可以使用 describe-export-tasks 命令來驗證任務已成功取消。

將 CloudWatch Logs 資料串流到 Amazon OpenSearch Service

您可以在 Amazon CloudWatch Logs 中設定日誌群組，因此您可以近乎即時地將資料串流至 Amazon OpenSearch Service 叢集。如需詳細資訊，請參閱使用訂閱即時處理日誌資料。

> **Note**
>
> 只有標準日誌類別中的日誌群組才支援串流至 OpenSearch Service。如需日誌類別的詳細資訊，請參閱 日誌類別。

根據串流的日誌資料量，請考慮設定函數層級並行限制。如需詳細資訊，請參閱 Lambda 函數擴展。

> **Note**
>
> 由於將大量 CloudWatch Logs 資料串流至 OpenSearch Service 可能會導致高使用費，我們建議您在 AWS 帳單與成本管理 主控台中建立預算。如需詳細資訊，請參閱使用 AWS Budgets 管理您的成本。

本節說明在將日誌群組訂閱至 OpenSearch Service 之前，您必須完成的先決條件。它還描述了如何將日誌群組訂閱到 OpenSearch Service。

先決條件

開始之前，請建立 OpenSearch Service 網域。網域可以有公有存取或 VPC 存取，但您不能在建立網域之後再修改存取類型。您之後可能會想要檢視 OpenSearch Service 網域設定，並根據叢集將會處理的資料量來修改叢集組態。如需有關建立網域的指示，請參閱建立 OpenSearch Service 網域。

如需有關 OpenSearch Service 的詳細資訊，請參閱《Amazon OpenSearch Service 開發人員指南》。

將日誌群組訂閱到 OpenSearch Service

您可以使用 CloudWatch 主控台將日誌群組訂閱到 OpenSearch Service。

將日誌群組訂閱到 OpenSearch Service

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Log groups (日誌群組)。
3. 選取日誌群組的名稱。
4. 選擇 Actions (動作)、Subscription filters (訂閱篩選條件)、Create Amazon OpenSearch Service subscription filter (建立 Amazon OpenSearch Service 訂閱篩選條件)。
5. 選擇您是否要串流至此帳戶或其他帳戶中的叢集。
   - 如果您選擇此帳戶，請選取您在前一個步驟所建立的網域。
   - 如果您選擇其他帳戶，請提供網域 ARN 和端點。
6. 針對 Lambda IAM Execution Role (Lambda IAM 執行角色)，選擇對 OpenSearch 執行呼叫時 Lambda 應該使用的 IAM 角色。

   您選擇的 IAM 角色必須符合這些要求：

   - 它必須擁有 `lambda.amazonaws.com` 的信任關係。
   - 它必須包含以下政策：

     JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOpenSearchStreamingAccess",
            "Action": [
                "es:*"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:es:us-
east-1:123456789012:domain/cloudwatch-logs/*"
        }
    ]
}
```

   - 如果目標 OpenSearch Service 網域使用 VPC 存取，該角色必須已連接 AWSLambdaVPCAccessExecutionRole 政策。此 Amazon 受管政策可授權 Lambda 存取客戶的 VPC，讓 Lambda 寫入 VPC 中的 OpenSearch 端點。

7. 針對 Log format (日誌格式)，選擇日誌格式。
8. 針對 Subscription filter pattern (訂閱篩選條件模式)，輸入要在您的日誌事件中尋找的詞彙或模式。這可確保您只會將所需的資料傳送至 OpenSearch 叢集。如需詳細資訊，請參閱使用篩選條件從日誌事件建立指標。
9. (選用) 針對 Select log data to test (選取要測試的日誌資料)，選擇一個日誌串流，然後選擇 Test pattern (測試模式)，以驗證您的搜尋篩選條件是否會傳回您預期的結果。
10. 選擇 Start streaming (開始串流)。

使用 SDK 的 CloudWatch Logs 程式碼範例 AWS SDKs

下列程式碼範例示範如何使用 CloudWatch Logs 搭配 AWS 軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- 建立您的第一個 Lambda 函數
- 使用 CloudWatch Logs 執行大型查詢
- 使用排程事件來調用 Lambda 函式

使用 SDK 的 CloudWatch Logs 基本範例 AWS SDKs

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon CloudWatch Logs。

範例

- 使用 SDK 的 CloudWatch Logs 動作 AWS SDKs
  - AssociateKmsKey 搭配 AWS SDK 使用
  - CancelExportTask 搭配 AWS SDK 使用
  - CreateExportTask 搭配 AWS SDK 使用
  - CreateLogGroup 搭配 AWS SDK 或 CLI 使用
  - CreateLogStream 搭配 AWS SDK 或 CLI 使用
  - DeleteLogGroup 搭配 AWS SDK 或 CLI 使用
  - DeleteSubscriptionFilter 搭配 AWS SDK 使用
  - DescribeExportTasks 搭配 AWS SDK 使用
  - DescribeLogGroups 搭配 AWS SDK 或 CLI 使用
  - DescribeLogStreams 搭配 AWS SDK 或 CLI 使用
  - DescribeSubscriptionFilters 搭配 AWS SDK 使用
  - GetLogEvents 搭配 AWS SDK 或 CLI 使用
  - GetQueryResults 搭配 AWS SDK 使用
  - PutSubscriptionFilter 搭配 AWS SDK 使用
  - StartLiveTail 搭配 AWS SDK 使用
  - StartQuery 搭配 AWS SDK 使用

使用 SDK 的 CloudWatch Logs 動作 AWS SDKs

下列程式碼範例示範如何使用 AWS SDKs 執行個別 CloudWatch Logs 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 CloudWatch Logs API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 使用 SDK 的 CloudWatch Logs 案例 AWS SDKs 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《Amazon CloudWatch Logs API 參考》。

範例

- AssociateKmsKey 搭配 AWS SDK 使用
- CancelExportTask 搭配 AWS SDK 使用
- CreateExportTask 搭配 AWS SDK 使用
- CreateLogGroup 搭配 AWS SDK 或 CLI 使用
- CreateLogStream 搭配 AWS SDK 或 CLI 使用
- DeleteLogGroup 搭配 AWS SDK 或 CLI 使用
- DeleteSubscriptionFilter 搭配 AWS SDK 使用
- DescribeExportTasks 搭配 AWS SDK 使用
- DescribeLogGroups 搭配 AWS SDK 或 CLI 使用
- DescribeLogStreams 搭配 AWS SDK 或 CLI 使用
- DescribeSubscriptionFilters 搭配 AWS SDK 使用
- GetLogEvents 搭配 AWS SDK 或 CLI 使用
- GetQueryResults 搭配 AWS SDK 使用
- PutSubscriptionFilter 搭配 AWS SDK 使用
- StartLiveTail 搭配 AWS SDK 使用
- StartQuery 搭配 AWS SDK 使用

`AssociateKmsKey` 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 `AssociateKmsKey`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    using System;
    using System.Threading.Tasks;
    using Amazon.CloudWatchLogs;
    using Amazon.CloudWatchLogs.Model;

    /// <summary>
    /// Shows how to associate an AWS Key Management Service (AWS KMS) key with
    /// an Amazon CloudWatch Logs log group.
    /// </summary>
    public class AssociateKmsKey
    {
        public static async Task Main()
        {
            // This client object will be associated with the same AWS Region
            // as the default user on this system. If you need to use a
            // different AWS Region, pass it as a parameter to the client
            // constructor.
            var client = new AmazonCloudWatchLogsClient();

            string kmsKeyId = "arn:aws:kms:us-west-2:<account-
number>:key/7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";
            string groupName = "cloudwatchlogs-example-loggroup";

            var request = new AssociateKmsKeyRequest
            {
                KmsKeyId = kmsKeyId,
                LogGroupName = groupName,
            };

            var response = await client.AssociateKmsKeyAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
```

```
            {
                Console.WriteLine($"Successfully associated KMS key ID:
{kmsKeyId} with log group: {groupName}.");
            }
            else
            {
                Console.WriteLine("Could not make the association between:
{kmsKeyId} and {groupName}.");
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 AssociateKmsKey。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CancelExportTask` 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 `CancelExportTask`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to cancel an Amazon CloudWatch Logs export task.
```

```
    /// </summary>
    public class CancelExportTask
    {
        public static async Task Main()
        {
            // This client object will be associated with the same AWS Region
            // as the default user on this system. If you need to use a
            // different AWS Region, pass it as a parameter to the client
            // constructor.
            var client = new AmazonCloudWatchLogsClient();
            string taskId = "exampleTaskId";

            var request = new CancelExportTaskRequest
            {
                TaskId = taskId,
            };

            var response = await client.CancelExportTaskAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"{taskId} successfully canceled.");
            }
            else
            {
                Console.WriteLine($"{taskId} could not be canceled.");
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 CancelExportTask。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateExportTask` 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 `CreateExportTask`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    using System;
    using System.Threading.Tasks;
    using Amazon.CloudWatchLogs;
    using Amazon.CloudWatchLogs.Model;

    /// <summary>
    /// Shows how to create an Export Task to export the contents of the Amazon
    /// CloudWatch Logs to the specified Amazon Simple Storage Service (Amazon
S3)
    /// bucket.
    /// </summary>
    public class CreateExportTask
    {
        public static async Task Main()
        {
            // This client object will be associated with the same AWS Region
            // as the default user on this system. If you need to use a
            // different AWS Region, pass it as a parameter to the client
            // constructor.
            var client = new AmazonCloudWatchLogsClient();
            string taskName = "export-task-example";
            string logGroupName = "cloudwatchlogs-example-loggroup";
            string destination = "amzn-s3-demo-bucket";
            var fromTime = 1437584472382;
            var toTime = 1437584472833;

            var request = new CreateExportTaskRequest
            {
                From = fromTime,
                To = toTime,
                TaskName = taskName,
                LogGroupName = logGroupName,
```

```
                Destination = destination,
            };

            var response = await client.CreateExportTaskAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"The task, {taskName} with ID: " +
                                  $"{response.TaskId} has been created
successfully.");
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱適用於 .NET 的 AWS SDK API 參考中的 CreateExportTask。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateLogGroup` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateLogGroup`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 設定 Amazon ECS Service Connect

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
```

```
    using System.Threading.Tasks;
    using Amazon.CloudWatchLogs;
    using Amazon.CloudWatchLogs.Model;

    /// <summary>
    /// Shows how to create an Amazon CloudWatch Logs log group.
    /// </summary>
    public class CreateLogGroup
    {
        public static async Task Main()
        {
            // This client object will be associated with the same AWS Region
            // as the default user on this system. If you need to use a
            // different AWS Region, pass it as a parameter to the client
            // constructor.
            var client = new AmazonCloudWatchLogsClient();

            string logGroupName = "cloudwatchlogs-example-loggroup";

            var request = new CreateLogGroupRequest
            {
                LogGroupName = logGroupName,
            };

            var response = await client.CreateLogGroupAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"Successfully create log group with ID:
{logGroupName}.");
            }
            else
            {
                Console.WriteLine("Could not create log group.");
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 CreateLogGroup。

CLI

AWS CLI

以下命令會建立名為 my-logs 的日誌群組：

```
aws logs create-log-group --log-group-name my-logs
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateLogGroup。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { CreateLogGroupCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
  const command = new CreateLogGroupCommand({
    // The name of the log group.
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
};

export default run();
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateLogGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateLogStream` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateLogStream`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to create an Amazon CloudWatch Logs stream for a CloudWatch
/// log group.
/// </summary>
public class CreateLogStream
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();
        string logGroupName = "cloudwatchlogs-example-loggroup";
        string logStreamName = "cloudwatchlogs-example-logstream";

        var request = new CreateLogStreamRequest
        {
            LogGroupName = logGroupName,
            LogStreamName = logStreamName,
```

```
            };

            var response = await client.CreateLogStreamAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"{logStreamName} successfully created for
{logGroupName}.");
            }
            else
            {
                Console.WriteLine("Could not create stream.");
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 CreateLogStream。

CLI

AWS CLI

以下範例會在日誌群組 my-logs 中建立名為 20150601 的日誌串流。

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateLogStream。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteLogGroup 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteLogGroup。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 設定 Amazon ECS Service Connect
- 建立您的第一個 Lambda 函數

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Uses the Amazon CloudWatch Logs Service to delete an existing
/// CloudWatch Logs log group.
/// </summary>
public class DeleteLogGroup
{
    public static async Task Main()
    {
        var client = new AmazonCloudWatchLogsClient();
        string logGroupName = "cloudwatchlogs-example-loggroup";

        var request = new DeleteLogGroupRequest
        {
            LogGroupName = logGroupName,
        };

        var response = await client.DeleteLogGroupAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"Successfully deleted CloudWatch log group,
{logGroupName}.");
        }
```

```
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 DeleteLogGroup。

CLI

AWS CLI

下列命令會刪除名為 my-logs 的日誌群組：

```
aws logs delete-log-group --log-group-name my-logs
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteLogGroup。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteLogGroupCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
  const command = new DeleteLogGroupCommand({
    // The name of the log group.
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
```

```
};

export default run();
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteLogGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteSubscriptionFilter` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `DeleteSubscriptionFilter`。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

包括必需的檔案。

```
#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/DeleteSubscriptionFilterRequest.h>
#include <iostream>
```

刪除訂閱篩選條件。

```
        Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
        Aws::CloudWatchLogs::Model::DeleteSubscriptionFilterRequest request;
        request.SetFilterName(filter_name);
```

```
        request.SetLogGroupName(log_group);

        auto outcome = cwl.DeleteSubscriptionFilter(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Failed to delete CloudWatch log subscription filter "
                << filter_name << ": " << outcome.GetError().GetMessage() <<
                std::endl;
        } else {
            std::cout << "Successfully deleted CloudWatch logs subscription " <<
                "filter " << filter_name << std::endl;
        }
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 DeleteSubscriptionFilter。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.DeleteSubscriptionFilterRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteSubscriptionFilter {
```

```
   public static void main(String[] args) {
       final String usage = """

               Usage:
                 <filter> <logGroup>

               Where:
                 filter - The name of the subscription filter (for example,
MyFilter).
                 logGroup - The name of the log group. (for example, testgroup).
               """;

       if (args.length != 2) {
           System.out.println(usage);
           System.exit(1);
       }

       String filter = args[0];
       String logGroup = args[1];
       CloudWatchLogsClient logs = CloudWatchLogsClient.builder()
               .build();

       deleteSubFilter(logs, filter, logGroup);
       logs.close();
   }

   public static void deleteSubFilter(CloudWatchLogsClient logs, String filter,
String logGroup) {
       try {
           DeleteSubscriptionFilterRequest request =
DeleteSubscriptionFilterRequest.builder()
                   .filterName(filter)
                   .logGroupName(logGroup)
                   .build();

           logs.deleteSubscriptionFilter(request);
           System.out.printf("Successfully deleted CloudWatch logs subscription
filter %s", filter);

       } catch (CloudWatchException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

```
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DeleteSubscriptionFilter。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteSubscriptionFilterCommand } from "@aws-sdk/client-cloudwatch-
logs";
import { client } from "../libs/client.js";

const run = async () => {
  const command = new DeleteSubscriptionFilterCommand({
    // The name of the filter.
    filterName: process.env.CLOUDWATCH_LOGS_FILTER_NAME,
    // The name of the log group.
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
};

export default run();
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteSubscriptionFilter。

適用於 JavaScript (v2) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
  filterName: "FILTER",
  logGroupName: "LOG_GROUP",
};

cwl.deleteSubscriptionFilter(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteSubscriptionFilter。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun deleteSubFilter(
    filter: String?,
    logGroup: String?,
) {
    val request =
        DeleteSubscriptionFilterRequest {
            filterName = filter
            logGroupName = logGroup
        }

    CloudWatchLogsClient.fromEnvironment { region = "us-west-2" }.use { logs ->
        logs.deleteSubscriptionFilter(request)
        println("Successfully deleted CloudWatch logs subscription filter named
 $filter")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的AWS SDK API 參考》中的 DeleteSubscriptionFilter。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeExportTasks` 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 `DescribeExportTasks`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to retrieve a list of information about Amazon CloudWatch
/// Logs export tasks.
/// </summary>
public class DescribeExportTasks
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();

        var request = new DescribeExportTasksRequest
        {
            Limit = 5,
        };

        var response = new DescribeExportTasksResponse();

        do
        {
            response = await client.DescribeExportTasksAsync(request);
            response.ExportTasks.ForEach(t =>
            {
```

```
                    Console.WriteLine($"{t.TaskName} with ID: {t.TaskId} has
status: {t.Status}");
                });
            }
            while (response.NextToken is not null);
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 DescribeExportTasks。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeLogGroups` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DescribeLogGroups`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Retrieves information about existing Amazon CloudWatch Logs log groups
/// and displays the information on the console.
/// </summary>
public class DescribeLogGroups
{
```

```
        public static async Task Main()
        {
            // Creates a CloudWatch Logs client using the default
            // user. If you need to work with resources in another
            // AWS Region than the one defined for the default user,
            // pass the AWS Region as a parameter to the client constructor.
            var client = new AmazonCloudWatchLogsClient();

            bool done = false;
            string newToken = null;

            var request = new DescribeLogGroupsRequest
            {
                Limit = 5,
            };

            DescribeLogGroupsResponse response;

            do
            {
                if (newToken is not null)
                {
                    request.NextToken = newToken;
                }

                response = await client.DescribeLogGroupsAsync(request);

                response.LogGroups.ForEach(lg =>
                {
                    Console.WriteLine($"{lg.LogGroupName} is associated with the
key: {lg.KmsKeyId}.");
                    Console.WriteLine($"Created on:
{lg.CreationTime.Date.Date}");
                    Console.WriteLine($"Date for this group will be stored for:
{lg.RetentionInDays} days.\n");
                });

                if (response.NextToken is null)
                {
                    done = true;
                }
                else
                {
                    newToken = response.NextToken;
```

```
                }
            }
            while (!done);
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 DescribeLogGroups。

CLI

AWS CLI

下列命令描述名為 my-logs 日誌群組：

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

輸出：

```
{
    "logGroups": [
        {
            "storedBytes": 0,
            "metricFilterCount": 0,
            "creationTime": 1433189500783,
            "logGroupName": "my-logs",
            "retentionInDays": 5,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeLogGroups。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import {
  paginateDescribeLogGroups,
  CloudWatchLogsClient,
} from "@aws-sdk/client-cloudwatch-logs";

const client = new CloudWatchLogsClient({});

export const main = async () => {
  const paginatedLogGroups = paginateDescribeLogGroups({ client }, {});
  const logGroups = [];

  for await (const page of paginatedLogGroups) {
    if (page.logGroups?.every((lg) => !!lg)) {
      logGroups.push(...page.logGroups);
    }
  }

  console.log(logGroups);
  return logGroups;
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DescribeLogGroups。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeLogStreams` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DescribeLogStreams`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立您的第一個 Lambda 函數

CLI

AWS CLI

下列命令顯示日誌群組 my-logs中以 2015 開頭的所有日誌串流：

```
aws logs describe-log-streams --log-group-name my-logs --log-stream-name-
prefix 2015
```

輸出：

```
{
    "logStreams": [
        {
            "creationTime": 1433189871774,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-
stream:20150531",
            "logStreamName": "20150531",
            "storedBytes": 0
        },
        {
            "creationTime": 1433189873898,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-
stream:20150601",
            "logStreamName": "20150601",
            "storedBytes": 0
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeLogStreams。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在符合指定字首的指定日誌群組內搜尋日誌串流。

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CloudWatchLogsSearch {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <logGroupName> <logStreamName>

                Where:
                  logGroupName - The name of the log group (for example,
 WeathertopJavaContainerLogs).
                  logStreamName - The name of the log stream (for example,
 weathertop-java-stream).
                  pattern - the pattern to use (for example, INFO)

                """;

        if (args.length != 3) {
            System.out.print(usage);
            System.exit(1);
        }
```

```
        String logGroupName = args[0] ;
        String logStreamName = args[1] ;
        String pattern = args[2] ;

        CloudWatchLogsClient cwlClient = CloudWatchLogsClient.builder()
                .region(Region.US_EAST_1)
                .build();

        searchLogStreamsAndFilterEvents(cwlClient, logGroupName, logStreamName,
pattern);
    }

    /**
     * Searches for log streams with a specific prefix within a log group and
filters log events based on a specified pattern.
     *
     * @param cwlClient       the CloudWatchLogsClient used to interact with AWS
CloudWatch Logs
     * @param logGroupName    the name of the log group to search within
     * @param logStreamPrefix the prefix of the log streams to search for
     * @param pattern         the pattern to filter log events by
     */
    public static void searchLogStreamsAndFilterEvents(CloudWatchLogsClient
cwlClient, String logGroupName, String logStreamPrefix, String pattern) {
        DescribeLogStreamsRequest describeLogStreamsRequest =
DescribeLogStreamsRequest.builder()
                .logGroupName(logGroupName)
                .logStreamNamePrefix(logStreamPrefix)
                .build();

        DescribeLogStreamsResponse describeLogStreamsResponse =
cwlClient.describeLogStreams(describeLogStreamsRequest);
        List<LogStream> logStreams = describeLogStreamsResponse.logStreams();

        for (LogStream logStream : logStreams) {
            String logStreamName = logStream.logStreamName();
            System.out.println("Searching in log stream: " + logStreamName);

            FilterLogEventsRequest filterLogEventsRequest =
FilterLogEventsRequest.builder()
                    .logGroupName(logGroupName)
                    .logStreamNames(logStreamName)
                    .filterPattern(pattern)
```

```
                    .build();

            FilterLogEventsResponse filterLogEventsResponse =
 cwlClient.filterLogEvents(filterLogEventsRequest);

            for (FilteredLogEvent event : filterLogEventsResponse.events()) {
                System.out.println(event.message());
            }


 System.out.println("-------------------------------------------------"); //
 Separator for better readability
        }
    }
}
```

列印指定日誌群組中最新日誌串流的中繼資料。

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CloudWatchLogQuery {
    public static void main(final String[] args) {
        final String usage = """
                Usage:
                  <logGroupName>

                Where:
                  logGroupName - The name of the log group (for example, /aws/
lambda/ChatAIHandler).
                """;

        if (args.length != 1) {
            System.out.print(usage);
            System.exit(1);
```

```
        }

        String logGroupName = "/aws/lambda/ChatAIHandler" ; //args[0];
        CloudWatchLogsClient logsClient = CloudWatchLogsClient.builder()
                .region(Region.US_EAST_1)
                .build();

        describeMostRecentLogStream(logsClient, logGroupName);
    }

    /**
     * Describes and prints metadata about the most recent log stream in the
specified log group.
     *
     * @param logsClient   the CloudWatchLogsClient used to interact with AWS
CloudWatch Logs
     * @param logGroupName the name of the log group
     */
    public static void describeMostRecentLogStream(CloudWatchLogsClient
logsClient, String logGroupName) {
        DescribeLogStreamsRequest streamsRequest =
DescribeLogStreamsRequest.builder()
                .logGroupName(logGroupName)
                .orderBy(OrderBy.LAST_EVENT_TIME)
                .descending(true)
                .limit(1)
                .build();

        try {
            DescribeLogStreamsResponse streamsResponse =
logsClient.describeLogStreams(streamsRequest);
            List<LogStream> logStreams = streamsResponse.logStreams();

            if (logStreams.isEmpty()) {
                System.out.println("No log streams found for log group: " +
logGroupName);
                return;
            }

            LogStream stream = logStreams.get(0);
            System.out.println("Most Recent Log Stream:");
            System.out.println("  Name: " + stream.logStreamName());
            System.out.println("  ARN: " + stream.arn());
            System.out.println("  Creation Time: " + stream.creationTime());
```

```
            System.out.println("  First Event Time: " +
 stream.firstEventTimestamp());
            System.out.println("  Last Event Time: " +
 stream.lastEventTimestamp());
            System.out.println("  Stored Bytes: " + stream.storedBytes());
            System.out.println("  Upload Sequence Token: " +
 stream.uploadSequenceToken());

        } catch (CloudWatchLogsException e) {
            System.err.println("Failed to describe log stream: " +
 e.awsErrorDetails().errorMessage());
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DescribeLogStreams。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeSubscriptionFilters` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `DescribeSubscriptionFilters`。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

包括必需的檔案。

```
#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/DescribeSubscriptionFiltersRequest.h>
```

```
#include <aws/logs/model/DescribeSubscriptionFiltersResult.h>
#include <iostream>
#include <iomanip>
```

列出訂閱篩選條件。

```
        Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
        Aws::CloudWatchLogs::Model::DescribeSubscriptionFiltersRequest request;
        request.SetLogGroupName(log_group);
        request.SetLimit(1);

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = cwl.DescribeSubscriptionFilters(
                    request);
            if (!outcome.IsSuccess()) {
                std::cout << "Failed to describe CloudWatch subscription filters
 "
                    << "for log group " << log_group << ": " <<
                    outcome.GetError().GetMessage() << std::endl;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(32) << "Name" <<
                    std::setw(64) << "FilterPattern" << std::setw(64) <<
                    "DestinationArn" << std::endl;
                header = true;
            }

            const auto &filters = outcome.GetResult().GetSubscriptionFilters();
            for (const auto &filter : filters) {
                std::cout << std::left << std::setw(32) <<
                    filter.GetFilterName() << std::setw(64) <<
                    filter.GetFilterPattern() << std::setw(64) <<
                    filter.GetDestinationArn() << std::endl;
            }

            const auto &next_token = outcome.GetResult().GetNextToken();
            request.SetNextToken(next_token);
            done = next_token.empty();
```

```
        }
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 DescribeSubscriptionFilters。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.DescribeSubscriptionFiltersRequest;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.DescribeSubscriptionFiltersResponse
import software.amazon.awssdk.services.cloudwatchlogs.model.SubscriptionFilter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeSubscriptionFilters {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                  <logGroup>
```

```
                Where:
                  logGroup - A log group name (for example, myloggroup).
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String logGroup = args[0];
        CloudWatchLogsClient logs = CloudWatchLogsClient.builder()
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        describeFilters(logs, logGroup);
        logs.close();
    }

    public static void describeFilters(CloudWatchLogsClient logs, String
logGroup) {
        try {
            boolean done = false;
            String newToken = null;

            while (!done) {
                DescribeSubscriptionFiltersResponse response;
                if (newToken == null) {
                    DescribeSubscriptionFiltersRequest request =
DescribeSubscriptionFiltersRequest.builder()
                            .logGroupName(logGroup)
                            .limit(1).build();

                    response = logs.describeSubscriptionFilters(request);
                } else {
                    DescribeSubscriptionFiltersRequest request =
DescribeSubscriptionFiltersRequest.builder()
                            .nextToken(newToken)
                            .logGroupName(logGroup)
                            .limit(1).build();
                    response = logs.describeSubscriptionFilters(request);
                }
```

```
                for (SubscriptionFilter filter : response.subscriptionFilters())
 {
                    System.out.printf("Retrieved filter with name %s, " +
 "pattern %s " + "and destination arn %s",
                            filter.filterName(),
                            filter.filterPattern(),
                            filter.destinationArn());
                }

                if (response.nextToken() == null) {
                    done = true;
                } else {
                    newToken = response.nextToken();
                }
            }

        } catch (CloudWatchException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.printf("Done");
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DescribeSubscriptionFilters。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DescribeSubscriptionFiltersCommand } from "@aws-sdk/client-cloudwatch-
logs";
import { client } from "../libs/client.js";
```

```
const run = async () => {
  // This will return a list of all subscription filters in your account
  // matching the log group name.
  const command = new DescribeSubscriptionFiltersCommand({
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
    limit: 1,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
};

export default run();
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DescribeSubscriptionFilters。

適用於 JavaScript (v2) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
  logGroupName: "GROUP_NAME",
  limit: 5,
};
```

```
cwl.describeSubscriptionFilters(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.subscriptionFilters);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DescribeSubscriptionFilters。

Kotlin

適用於 Kotlin 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun describeFilters(logGroup: String) {
    val request =
        DescribeSubscriptionFiltersRequest {
            logGroupName = logGroup
            limit = 1
        }

    CloudWatchLogsClient.fromEnvironment { region = "us-west-2" }.use { cwlClient
 ->
        val response = cwlClient.describeSubscriptionFilters(request)
        response.subscriptionFilters?.forEach { filter ->
            println("Retrieved filter with name  ${filter.filterName} pattern
 ${filter.filterPattern} and destination ${filter.destinationArn}")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的AWS SDK API 參考》中的 DescribeSubscriptionFilters。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetLogEvents` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetLogEvents`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立您的第一個 Lambda 函數

CLI

AWS CLI

下列命令會從日誌群組 `my-logs` 中名為 `20150601` 的日誌串流擷取日誌事件：

```
aws logs get-log-events --log-group-name my-logs --log-stream-name 20150601
```

輸出：

```
{
    "nextForwardToken":
 "f/31961209122447488583055879464742346735121166569214640130",
    "events": [
        {
            "ingestionTime": 1433190494190,
            "timestamp": 1433190184356,
            "message": "Example Event 1"
        },
        {
            "ingestionTime": 1433190516679,
            "timestamp": 1433190184356,
            "message": "Example Event 1"
        },
        {
            "ingestionTime": 1433190494190,
```

```
            "timestamp": 1433190184358,
            "message": "Example Event 2"
        }
    ],
    "nextBackwardToken":
 "b/319612091223582856022617569449886743245533732682167091 20"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetLogEvents。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.DescribeLogStreamsRequest;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.DescribeLogStreamsResponse;
import software.amazon.awssdk.services.cloudwatchlogs.model.GetLogEventsRequest;
import software.amazon.awssdk.services.cloudwatchlogs.model.GetLogEventsResponse;

import java.time.Instant;
import java.time.temporal.ChronoUnit;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
```

```
public class GetLogEvents {

    public static void main(String[] args) {

        final String usage = """

                Usage:
                  <logGroupName> <logStreamName>

                Where:
                  logGroupName - The name of the log group (for example,
 myloggroup).
                  logStreamName - The name of the log stream (for example,
 mystream).

                """;

       // if (args.length != 2) {
       //     System.out.print(usage);
       //     System.exit(1);
//        }

        String logGroupName = "WeathertopJavaContainerLogs" ; //args[0];
        String logStreamName = "weathertop-java-stream" ; //args[1];

        Region region = Region.US_EAST_1 ;
        CloudWatchLogsClient cloudWatchLogsClient =
 CloudWatchLogsClient.builder()
                .region(region)
                .build();

        getCWLogEvents(cloudWatchLogsClient, logGroupName, logStreamName);
        cloudWatchLogsClient.close();
    }

    public static void getCWLogEvents(CloudWatchLogsClient cloudWatchLogsClient,
                                      String logGroupName,
                                      String logStreamPrefix) {
        try {
            // First, find the exact log stream name
            DescribeLogStreamsRequest describeRequest =
 DescribeLogStreamsRequest.builder()
                    .logGroupName(logGroupName)
                    .logStreamNamePrefix(logStreamPrefix)
```

```
                    .limit(1) // get the first matching stream
                    .build();

            DescribeLogStreamsResponse describeResponse =
cloudWatchLogsClient.describeLogStreams(describeRequest);

            if (describeResponse.logStreams().isEmpty()) {
                System.out.println("No matching log streams found for prefix: " +
logStreamPrefix);
                return;
            }

            String exactLogStreamName =
describeResponse.logStreams().get(0).logStreamName();
            System.out.println("Using exact log stream: " + exactLogStreamName);

            long startTime = Instant.now().minus(7,
ChronoUnit.DAYS).toEpochMilli();
            long endTime = Instant.now().toEpochMilli();

            GetLogEventsRequest getLogEventsRequest =
GetLogEventsRequest.builder()
                    .logGroupName(logGroupName)
                    .logStreamName(exactLogStreamName) // <-- exact name, not
prefix
                    .startTime(startTime)
                    .endTime(endTime)
                    .startFromHead(true)
                    .build();

            GetLogEventsResponse response =
cloudWatchLogsClient.getLogEvents(getLogEventsRequest);

            if (response.events().isEmpty()) {
                System.out.println("No log events found in the past 7 days.");
            } else {
                response.events().forEach(e -> System.out.println(e.message()));
            }

        } catch (CloudWatchException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 GetLogEvents。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetQueryResults` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `GetQueryResults`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 執行大型查詢

.NET

適用於 .NET 的 SDK (v4)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Gets the results of a CloudWatch Logs Insights query.
    /// </summary>
    /// <param name="queryId">The ID of the query.</param>
    /// <returns>The query results response.</returns>
    public async Task<GetQueryResultsResponse?> GetQueryResultsAsync(string
queryId)
    {
        try
        {
            var request = new GetQueryResultsRequest
            {
                QueryId = queryId
```

```
            };

            var response = await
_amazonCloudWatchLogs.GetQueryResultsAsync(request);
            return response;
        }
        catch (ResourceNotFoundException ex)
        {
            _logger.LogError($"Query not found: {ex.Message}");
            return null;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while getting query results:
{ex.Message}");
            return null;
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 GetQueryResults。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
  return this.client.send(new GetQueryResultsCommand({ queryId }));
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetQueryResults。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def _wait_for_query_results(self, client, query_id):
    """
    Waits for the query to complete and retrieves the results.

    :param query_id: The ID of the initiated query.
    :type query_id: str
    :return: A list containing the results of the query.
    :rtype: list
    """
    while True:
        time.sleep(1)
        results = client.get_query_results(queryId=query_id)
        if results["status"] in [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ]:
            return results.get("results", [])
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 GetQueryResults。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_cwl->getqueryresults(
          iv_queryid = iv_query_id ).

        " Display query status and result count
        DATA(lv_status) = oo_result->get_status( ).
        DATA(lt_results) = oo_result->get_results( ).
        DATA(lv_result_count) = lines( lt_results ).

        MESSAGE |Query status: { lv_status }. Retrieved { lv_result_count } log
event(s).| TYPE 'I'.
      CATCH /aws1/cx_cwlinvalidparameterex.
        MESSAGE 'Invalid parameter.' TYPE 'E'.
      CATCH /aws1/cx_cwlresourcenotfoundex.
        MESSAGE 'Resource not found.' TYPE 'E'.
      CATCH /aws1/cx_cwlserviceunavailex.
        MESSAGE 'Service unavailable.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetQueryResults。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`PutSubscriptionFilter` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `PutSubscriptionFilter`。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

包括必需的檔案。

```
#include <aws/core/Aws.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/PutSubscriptionFilterRequest.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

建立訂閱篩選條件。

```
        Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
        Aws::CloudWatchLogs::Model::PutSubscriptionFilterRequest request;
        request.SetFilterName(filter_name);
        request.SetFilterPattern(filter_pattern);
        request.SetLogGroupName(log_group);
        request.SetDestinationArn(dest_arn);
        auto outcome = cwl.PutSubscriptionFilter(request);
        if (!outcome.IsSuccess())
        {
            std::cout << "Failed to create CloudWatch logs subscription filter "
                << filter_name << ": " << outcome.GetError().GetMessage() <<
                std::endl;
        }
        else
        {
            std::cout << "Successfully created CloudWatch logs subscription " <<
                "filter " << filter_name << std::endl;
        }
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的 PutSubscriptionFilter。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.CloudWatchLogsException;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.PutSubscriptionFilterRequest;

/**
 * Before running this code example, you need to grant permission to CloudWatch
 * Logs the right to execute your Lambda function.
 * To perform this task, you can use this CLI command:
 *
 * aws lambda add-permission --function-name "lamda1" --statement-id "lamda1"
 * --principal "logs.us-west-2.amazonaws.com" --action "lambda:InvokeFunction"
 * --source-arn "arn:aws:logs:us-west-2:111111111111:log-group:testgroup:*"
 * --source-account "111111111111"
 *
 * Make sure you replace the function name with your function name and replace
 * '111111111111' with your account details.
 * For more information, see "Subscription Filters with AWS Lambda" in the
 * Amazon CloudWatch Logs Guide.
 *
 *
 * Also, before running this Java V2 code example,set up your development
 * environment,including your credentials.
 *
 * For more information,see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 */

public class PutSubscriptionFilter {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <filter> <pattern> <logGroup> <functionArn>\s

                Where:
                  filter - A filter name (for example, myfilter).
                  pattern - A filter pattern (for example, ERROR).
                  logGroup - A log group name (testgroup).
                  functionArn - An AWS Lambda function ARN (for example,
 arn:aws:lambda:us-west-2:111111111111:function:lambda1) .
                """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String filter = args[0];
        String pattern = args[1];
        String logGroup = args[2];
        String functionArn = args[3];
        Region region = Region.US_WEST_2;
        CloudWatchLogsClient cwl = CloudWatchLogsClient.builder()
                .region(region)
                .build();

        putSubFilters(cwl, filter, pattern, logGroup, functionArn);
        cwl.close();
    }

    public static void putSubFilters(CloudWatchLogsClient cwl,
            String filter,
            String pattern,
            String logGroup,
            String functionArn) {
```

```
        try {
            PutSubscriptionFilterRequest request =
 PutSubscriptionFilterRequest.builder()
                    .filterName(filter)
                    .filterPattern(pattern)
                    .logGroupName(logGroup)
                    .destinationArn(functionArn)
                    .build();

            cwl.putSubscriptionFilter(request);
            System.out.printf(
                    "Successfully created CloudWatch logs subscription filter
 %s",
                    filter);

        } catch (CloudWatchLogsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 PutSubscriptionFilter。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { PutSubscriptionFilterCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
  const command = new PutSubscriptionFilterCommand({
    // An ARN of a same-account Kinesis stream, Kinesis Firehose
    // delivery stream, or Lambda function.
```

```
    // https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/
SubscriptionFilters.html
    destinationArn: process.env.CLOUDWATCH_LOGS_DESTINATION_ARN,

    // A name for the filter.
    filterName: process.env.CLOUDWATCH_LOGS_FILTER_NAME,

    // A filter pattern for subscribing to a filtered stream of log events.
    // https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/
FilterAndPatternSyntax.html
    filterPattern: process.env.CLOUDWATCH_LOGS_FILTER_PATTERN,

    // The name of the log group. Messages in this group matching the filter
 pattern
    // will be sent to the destination ARN.
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
};

export default run();
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 PutSubscriptionFilter。

適用於 JavaScript (v2) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });
```

```
// Create the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
  destinationArn: "LAMBDA_FUNCTION_ARN",
  filterName: "FILTER_NAME",
  filterPattern: "ERROR",
  logGroupName: "LOG_GROUP",
};

cwl.putSubscriptionFilter(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 PutSubscriptionFilter。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`StartLiveTail` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `StartLiveTail`。

.NET

適用於 .NET 的 SDK

包括必需的檔案。

```
using Amazon;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
```

開始 Live Tail 工作階段。

```
            var client = new AmazonCloudWatchLogsClient();
            var request = new StartLiveTailRequest
            {
                LogGroupIdentifiers = logGroupIdentifiers,
                LogStreamNames = logStreamNames,
                LogEventFilterPattern = filterPattern,
            };

            var response = await client.StartLiveTailAsync(request);

            // Catch if request fails
            if (response.HttpStatusCode != System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine("Failed to start live tail session");
                return;
            }
```

您可以透過兩種方式處理 Live Tail 工作階段中的事件：

```
            /* Method 1
            * 1). Asynchronously loop through the event stream
            * 2). Set a timer to dispose the stream and stop the Live Tail
session at the end.
            */
            var eventStream = response.ResponseStream;
            var task = Task.Run(() =>
            {
                foreach (var item in eventStream)
                {
                    if (item is LiveTailSessionUpdate liveTailSessionUpdate)
                    {
                        foreach (var sessionResult in
liveTailSessionUpdate.SessionResults)
                        {
                            Console.WriteLine("Message : {0}",
sessionResult.Message);
                        }
                    }
                    if (item is LiveTailSessionStart)
                    {
```

```
                    Console.WriteLine("Live Tail session started");
                }
                // On-stream exceptions are processed here
                if (item is CloudWatchLogsEventStreamException)
                {
                    Console.WriteLine($"ERROR: {item}");
                }
            }
        });
        // Close the stream to stop the session after a timeout
        if (!task.Wait(TimeSpan.FromSeconds(10))){
            eventStream.Dispose();
            Console.WriteLine("End of line");
        }
```

```
        /* Method 2
        * 1). Add event handlers to each event variable
        * 2). Start processing the stream and wait for a timeout using
AutoResetEvent
        */
        AutoResetEvent endEvent = new AutoResetEvent(false);
        var eventStream = response.ResponseStream;
        using (eventStream) // automatically disposes the stream to stop the
session after execution finishes
        {
            eventStream.SessionStartReceived += (sender, e) =>
            {
                Console.WriteLine("LiveTail session started");
            };
            eventStream.SessionUpdateReceived += (sender, e) =>
            {
                foreach (LiveTailSessionLogEvent logEvent in
e.EventStreamEvent.SessionResults){
                    Console.WriteLine("Message: {0}", logEvent.Message);
                }
            };
            // On-stream exceptions are captured here
            eventStream.ExceptionReceived += (sender, e) =>
            {
                Console.WriteLine($"ERROR:
{e.EventStreamException.Message}");
            };
```

```
                eventStream.StartProcessing();
                // Stream events for this amount of time.
                endEvent.WaitOne(TimeSpan.FromSeconds(10));
                Console.WriteLine("End of line");
            }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 StartLiveTail。

Go

SDK for Go V2

包括必需的檔案。

```
import (
 "context"
 "log"
 "time"

 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs/types"
)
```

處理 Live Tail 工作階段的事件。

```
func handleEventStreamAsync(stream *cloudwatchlogs.StartLiveTailEventStream) {
 eventsChan := stream.Events()
 for {
  event := <-eventsChan
  switch e := event.(type) {
  case *types.StartLiveTailResponseStreamMemberSessionStart:
   log.Println("Received SessionStart event")
  case *types.StartLiveTailResponseStreamMemberSessionUpdate:
   for _, logEvent := range e.Value.SessionResults {
    log.Println(*logEvent.Message)
   }
  default:
   // Handle on-stream exceptions
```

```
   if err := stream.Err(); err != nil {
    log.Fatalf("Error occured during streaming: %v", err)
   } else if event == nil {
    log.Println("Stream is Closed")
    return
   } else {
    log.Fatalf("Unknown event type: %T", e)
   }
  }
 }
}
```

開始 Live Tail 工作階段。

```
cfg, err := config.LoadDefaultConfig(context.TODO())
if err != nil {
 panic("configuration error, " + err.Error())
}
client := cloudwatchlogs.NewFromConfig(cfg)

request := &cloudwatchlogs.StartLiveTailInput{
 LogGroupIdentifiers:   logGroupIdentifiers,
 LogStreamNames:        logStreamNames,
 LogEventFilterPattern: logEventFilterPattern,
}

response, err := client.StartLiveTail(context.TODO(), request)
// Handle pre-stream Exceptions
if err != nil {
 log.Fatalf("Failed to start streaming: %v", err)
}

// Start a Goroutine to handle events over stream
stream := response.GetStream()
go handleEventStreamAsync(stream)
```

在經過一段時間後停止 Live Tail 工作階段。

```
// Close the stream (which ends the session) after a timeout
time.Sleep(10 * time.Second)
stream.Close()
```

```
 log.Println("Event stream closed")
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 StartLiveTail。

Java

適用於 Java 2.x 的 SDK

包括必需的檔案。

```
import io.reactivex.FlowableSubscriber;
import io.reactivex.annotations.NonNull;
import org.reactivestreams.Subscription;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsAsyncClient;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionLogEvent;
import software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionStart;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionUpdate;
import software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailRequest;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailResponseHandler;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.CloudWatchLogsException;
import
 software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailResponseStream;

import java.util.Date;
import java.util.List;
import java.util.concurrent.atomic.AtomicReference;
```

處理 Live Tail 工作階段的事件。

```
    private static StartLiveTailResponseHandler
 getStartLiveTailResponseStreamHandler(
            AtomicReference<Subscription> subscriptionAtomicReference) {
        return StartLiveTailResponseHandler.builder()
            .onResponse(r -> System.out.println("Received initial response"))
            .onError(throwable -> {
```

```
                CloudWatchLogsException e = (CloudWatchLogsException)
throwable.getCause();
                System.err.println(e.awsErrorDetails().errorMessage());
                System.exit(1);
            })
            .subscriber(() -> new FlowableSubscriber<>() {
                @Override
                public void onSubscribe(@NonNull Subscription s) {
                    subscriptionAtomicReference.set(s);
                    s.request(Long.MAX_VALUE);
                }

                @Override
                public void onNext(StartLiveTailResponseStream event) {
                    if (event instanceof LiveTailSessionStart) {
                        LiveTailSessionStart sessionStart =
(LiveTailSessionStart) event;
                        System.out.println(sessionStart);
                    } else if (event instanceof LiveTailSessionUpdate) {
                        LiveTailSessionUpdate sessionUpdate =
(LiveTailSessionUpdate) event;
                        List<LiveTailSessionLogEvent> logEvents =
sessionUpdate.sessionResults();
                        logEvents.forEach(e -> {
                            long timestamp = e.timestamp();
                            Date date = new Date(timestamp);
                            System.out.println("[" + date + "] " + e.message());
                        });
                    } else {
                        throw CloudWatchLogsException.builder().message("Unknown
event type").build();
                    }
                }

                @Override
                public void onError(Throwable throwable) {
                    System.out.println(throwable.getMessage());
                    System.exit(1);
                }

                @Override
                public void onComplete() {
                    System.out.println("Completed Streaming Session");
                }
```

```
            })
            .build();
    }
```

開始 Live Tail 工作階段。

```
        CloudWatchLogsAsyncClient cloudWatchLogsAsyncClient =
                CloudWatchLogsAsyncClient.builder()
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();

        StartLiveTailRequest request =
                StartLiveTailRequest.builder()
                    .logGroupIdentifiers(logGroupIdentifiers)
                    .logStreamNames(logStreamNames)
                    .logEventFilterPattern(logEventFilterPattern)
                    .build();

        /* Create a reference to store the subscription */
        final AtomicReference<Subscription> subscriptionAtomicReference = new
AtomicReference<>(null);

        cloudWatchLogsAsyncClient.startLiveTail(request,
getStartLiveTailResponseStreamHandler(subscriptionAtomicReference));
```

在經過一段時間後停止 Live Tail 工作階段。

```
        /* Set a timeout for the session and cancel the subscription. This will:
         * 1). Close the stream
         * 2). Stop the Live Tail session
         */
        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        if (subscriptionAtomicReference.get() != null) {
            subscriptionAtomicReference.get().cancel();
            System.out.println("Subscription to stream closed");
        }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 StartLiveTail。

JavaScript

適用於 JavaScript (v3) 的 SDK

包括必需的檔案。

```
import { CloudWatchLogsClient, StartLiveTailCommand } from "@aws-sdk/client-
cloudwatch-logs";
```

處理 Live Tail 工作階段的事件。

```
async function handleResponseAsync(response) {
    try {
      for await (const event of response.responseStream) {
        if (event.sessionStart !== undefined) {
          console.log(event.sessionStart);
        } else if (event.sessionUpdate !== undefined) {
          for (const logEvent of event.sessionUpdate.sessionResults) {
            const timestamp = logEvent.timestamp;
            const date = new Date(timestamp);
            console.log("[" + date + "] " + logEvent.message);
          }
        } else {
            console.error("Unknown event type");
        }
      }
    } catch (err) {
        // On-stream exceptions are captured here
        console.error(err)
    }
}
```

開始 Live Tail 工作階段。

```
    const client = new CloudWatchLogsClient();
```

```
    const command = new StartLiveTailCommand({
        logGroupIdentifiers: logGroupIdentifiers,
        logStreamNames: logStreamNames,
        logEventFilterPattern: filterPattern
    });
    try{
        const response = await client.send(command);
        handleResponseAsync(response);
    } catch (err){
        // Pre-stream exceptions are captured here
        console.log(err);
    }
```

在經過一段時間後停止 Live Tail 工作階段。

```
    /* Set a timeout to close the client. This will stop the Live Tail session.
*/
    setTimeout(function() {
        console.log("Client timeout");
        client.destroy();
      }, 10000);
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 StartLiveTail。

Kotlin

SDK for Kotlin

包括必需的檔案。

```
import aws.sdk.kotlin.services.cloudwatchlogs.CloudWatchLogsClient
import aws.sdk.kotlin.services.cloudwatchlogs.model.StartLiveTailRequest
import aws.sdk.kotlin.services.cloudwatchlogs.model.StartLiveTailResponseStream
import kotlinx.coroutines.flow.takeWhile
```

開始 Live Tail 工作階段。

```
    val client = CloudWatchLogsClient.fromEnvironment()

    val request = StartLiveTailRequest {
        logGroupIdentifiers = logGroupIdentifiersVal
        logStreamNames = logStreamNamesVal
        logEventFilterPattern = logEventFilterPatternVal
    }

    val startTime = System.currentTimeMillis()

    try {
        client.startLiveTail(request) { response ->
            val stream = response.responseStream
            if (stream != null) {
                /* Set a timeout to unsubcribe from the flow. This will:
                * 1). Close the stream
                * 2). Stop the Live Tail session
                */
                stream.takeWhile { System.currentTimeMillis() - startTime <
10000 }.collect { value ->
                    if (value is StartLiveTailResponseStream.SessionStart) {
                        println(value.asSessionStart())
                    } else if (value is
StartLiveTailResponseStream.SessionUpdate) {
                        for (e in value.asSessionUpdate().sessionResults!!) {
                            println(e)
                        }
                    } else {
                        throw IllegalArgumentException("Unknown event type")
                    }
                }
            } else {
                throw IllegalArgumentException("No response stream")
            }
        }
    } catch (e: Exception) {
        println("Exception occurred during StartLiveTail: $e")
        System.exit(1)
    }
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 StartLiveTail。

Python

適用於 Python 的 SDK (Boto3)

包括必需的檔案。

```
import boto3
import time
from datetime import datetime
```

開始 Live Tail 工作階段。

```
    # Initialize the client
    client = boto3.client('logs')

    start_time = time.time()

    try:
        response = client.start_live_tail(
            logGroupIdentifiers=log_group_identifiers,
            logStreamNames=log_streams,
            logEventFilterPattern=filter_pattern
        )
        event_stream = response['responseStream']
        # Handle the events streamed back in the response
        for event in event_stream:
            # Set a timeout to close the stream.
            # This will end the Live Tail session.
            if (time.time() - start_time >= 10):
                event_stream.close()
                break
            # Handle when session is started
            if 'sessionStart' in event:
                session_start_event = event['sessionStart']
                print(session_start_event)
            # Handle when log event is given in a session update
            elif 'sessionUpdate' in event:
                log_events = event['sessionUpdate']['sessionResults']
                for log_event in log_events:
                    print('[{date}]
 {log}'.format(date=datetime.fromtimestamp(log_event['timestamp']/1000),log=log_event['me
            else:
```

```
                # On-stream exceptions are captured here
                raise RuntimeError(str(event))
    except Exception as e:
        print(e)
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 StartLiveTail。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`StartQuery` 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 `StartQuery`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 執行大型查詢

.NET

適用於 .NET 的 SDK (v4)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Starts a CloudWatch Logs Insights query.
    /// </summary>
    /// <param name="logGroupName">The name of the log group to query.</param>
    /// <param name="queryString">The CloudWatch Logs Insights query string.</
param>
    /// <param name="startTime">The start time for the query (seconds since
 epoch).</param>
    /// <param name="endTime">The end time for the query (seconds since epoch).</
param>
```

```
    /// <param name="limit">The maximum number of results to return.</param>
    /// <returns>The query ID if successful, null otherwise.</returns>
    public async Task<string?> StartQueryAsync(
        string logGroupName,
        string queryString,
        long startTime,
        long endTime,
        int limit = 10000)
    {
        try
        {
            var request = new StartQueryRequest
            {
                LogGroupName = logGroupName,
                QueryString = queryString,
                StartTime = startTime,
                EndTime = endTime,
                Limit = limit
            };

            var response = await _amazonCloudWatchLogs.StartQueryAsync(request);
            return response.QueryId;
        }
        catch (InvalidParameterException ex)
        {
            _logger.LogError($"Invalid parameter for query: {ex.Message}");
            return null;
        }
        catch (ResourceNotFoundException ex)
        {
            _logger.LogError($"Log group not found: {ex.Message}");
            return null;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while starting query:
{ex.Message}");
            return null;
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 StartQuery。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/**
 * Wrapper for the StartQueryCommand. Uses a static query string
 * for consistency.
 * @param {[Date, Date]} dateRange
 * @param {number} maxLogs
 * @returns {Promise<{ queryId: string }>}
 */
async _startQuery([startDate, endDate], maxLogs = 10000) {
  try {
    return await this.client.send(
      new StartQueryCommand({
        logGroupNames: this.logGroupNames,
        queryString: "fields @timestamp, @message | sort @timestamp asc",
        startTime: startDate.valueOf(),
        endTime: endDate.valueOf(),
        limit: maxLogs,
      }),
    );
  } catch (err) {
    /** @type {string} */
    const message = err.message;
    if (message.startsWith("Query's end date and time")) {
      // This error indicates that the query's start or end date occur
      // before the log group was created.
      throw new DateOutOfBoundsError(message);
    }

    throw err;
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 StartQuery。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    def perform_query(self, date_range):
        """
        Performs the actual CloudWatch log query.

        :param date_range: A tuple representing the start and end datetime for
the query.
        :type date_range: tuple
        :return: A list containing the query results.
        :rtype: list
        """
        client = boto3.client("logs")
        try:
            try:
                start_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
                )
                end_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
                )
                response = client.start_query(
                    logGroupName=self.log_group,
                    startTime=start_time,
                    endTime=end_time,
                    queryString=self.query_string,
                    limit=self.limit,
                )
                query_id = response["queryId"]
            except client.exceptions.ResourceNotFoundException as e:
```

```
                raise DateOutOfBoundsError(f"Resource not found: {e}")
            while True:
                time.sleep(1)
                results = client.get_query_results(queryId=query_id)
                if results["status"] in [
                    "Complete",
                    "Failed",
                    "Cancelled",
                    "Timeout",
                    "Unknown",
                ]:
                    return results.get("results", [])
        except DateOutOfBoundsError:
            return []

    def _initiate_query(self, client, date_range, max_logs):
        """
        Initiates the CloudWatch logs query.

        :param date_range: A tuple representing the start and end datetime for
the query.
        :type date_range: tuple
        :param max_logs: The maximum number of logs to retrieve.
        :type max_logs: int
        :return: The query ID as a string.
        :rtype: str
        """
        try:
            start_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
            )
            end_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
            )
            response = client.start_query(
                logGroupName=self.log_group,
                startTime=start_time,
                endTime=end_time,
                queryString=self.query_string,
                limit=max_logs,
            )
            return response["queryId"]
```

```
        except client.exceptions.ResourceNotFoundException as e:
            raise DateOutOfBoundsError(f"Resource not found: {e}")
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 StartQuery。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
   TRY.
       " iv_log_group_name = '/aws/lambda/my-function'
       " iv_query_string = 'fields @timestamp, @message | sort @timestamp desc |
limit 20'
       " iv_start_time and iv_end_time must be in Unix epoch milliseconds (ms
since Jan 1, 1970 00:00:00 UTC)
       oo_result = lo_cwl->startquery(
         iv_loggroupname = iv_log_group_name
         iv_starttime    = iv_start_time
         iv_endtime      = iv_end_time
         iv_querystring  = iv_query_string
         iv_limit        = iv_limit ).

       " Display the query ID for tracking
       DATA(lv_query_id) = oo_result->get_queryid( ).
       MESSAGE |Query started successfully with ID: { lv_query_id }| TYPE 'I'.
     CATCH /aws1/cx_cwlinvalidparameterex.
       MESSAGE 'Invalid parameter.' TYPE 'E'.
     CATCH /aws1/cx_cwllimitexceededex.
       MESSAGE 'Limit exceeded.' TYPE 'E'.
     CATCH /aws1/cx_cwlmalformedqueryex.
       MESSAGE 'Malformed query.' TYPE 'E'.
     CATCH /aws1/cx_cwlresourcenotfoundex.
       MESSAGE 'Resource not found.' TYPE 'E'.
     CATCH /aws1/cx_cwlserviceunavailex.
```

```
        MESSAGE 'Service unavailable.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 StartQuery。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 的 CloudWatch Logs 案例 AWS SDKs

下列程式碼範例示範如何使用 AWS SDKs 在 CloudWatch Logs 中實作常見案例。這些案例示範如何呼叫 CloudWatch Logs 中的多個函數或與其他 AWS 服務組合來完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- 設定 Amazon ECS Service Connect
- 建立您的第一個 Lambda 函數
- 使用 CloudWatch Logs 執行大型查詢
- 使用排程事件來調用 Lambda 函式

設定 Amazon ECS Service Connect

以下程式碼範例顯示做法：

- 建立 VPC 基礎設施
- 設定記錄
- 建立 ECS 叢集
- 設定 IAM 角色
- 使用 Service Connect 建立服務
- 驗證部署
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
```

```
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"
```

```
    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"
```

```
    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}
```

```
# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx" 2>&1
 | grep -v "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy" 2>&1
 | grep -v "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
```

```
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
```

```
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
```

```
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
```

```
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        }
    ]
```

```
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
```

```
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================="
    echo "CREATED RESOURCES"
    echo "============================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================="
```

```
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
```

```
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
```

```
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"
```

```
    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster

- DeleteLogGroup
- DeleteNamespace
- DeleteRole
- DeleteSecurityGroup
- DeleteService
- DeregisterTaskDefinition
- DescribeClusters
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立您的第一個 Lambda 函數

以下程式碼範例顯示做法：

- 建立 Lambda 的 IAM 角色
- 建立函數程式碼
- 建立 Lambda 函式

- 測試您的 Lambda 函數
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
# views CloudWatch logs, and cleans up all resources.
#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE

###############################################################################
# Setup
###############################################################################

UNIQUE_ID=$(cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 8 | head -n 1)
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1
```

```
declare -a CREATED_RESOURCES

################################################################################
# Helper functions
################################################################################

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
        local NAME="${RESOURCE#*:}"

        case "$TYPE" in
            log-group)
                echo "Deleting CloudWatch log group: ${NAME}"
                aws logs delete-log-group \
                    --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
 delete log group ${NAME}."
                ;;
            lambda-function)
                echo "Deleting Lambda function: ${NAME}"
                aws lambda delete-function \
                    --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
 delete Lambda function ${NAME}."
                echo "  Waiting for function deletion to complete..."
                local DELETE_WAIT=0
                while aws lambda get-function --function-name "$NAME" > /dev/null
 2>&1; do
                    sleep 2
                    DELETE_WAIT=$((DELETE_WAIT + 2))
                    if [ "$DELETE_WAIT" -ge 60 ]; then
                        echo "  WARNING: Timed out waiting for function
 deletion."
                        break
                    fi
                done
```

```
                ;;
            iam-role-policy)
                local ROLE_PART="${NAME%%|*}"
                local POLICY_PART="${NAME#*|}"
                echo "Detaching policy from role: ${ROLE_PART}"
                aws iam detach-role-policy \
                    --role-name "$ROLE_PART" \
                    --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
 not detach policy from role ${ROLE_PART}."
                ;;
            iam-role)
                echo "Deleting IAM role: ${NAME}"
                aws iam delete-role \
                    --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
 IAM role ${NAME}."
                ;;
        esac
    done

    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "========================================="
    echo "ERROR: Script failed at $1"
    echo "========================================="
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
```

```
    local COMMAND="$2"
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
 seconds."
            return 1
        fi
        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

###############################################################################
# Region pre-check
###############################################################################

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
 "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi

###############################################################################
# Runtime selection
###############################################################################

echo ""
echo "=========================================="
echo "AWS Lambda - Create Your First Function"
echo "=========================================="
```

```
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Enter your choice (1 or 2): "
read -r RUNTIME_CHOICE

case "$RUNTIME_CHOICE" in
    1)
        RUNTIME="python3.13"
        HANDLER="lambda_function.lambda_handler"
        CODE_FILE="lambda_function.py"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'
import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    length = event['length']
    width = event['width']
    area = calculate_area(length, width)
    print(f'The area is {area}')
    logger.info(f'CloudWatch logs group: {context.log_group_name}')
    return json.dumps({'area': area})
def calculate_area(length, width):
    return length * width
PYTHON_EOF
        echo "Selected runtime: Python 3.13"
        ;;
    2)
        RUNTIME="nodejs22.x"
        HANDLER="index.handler"
        CODE_FILE="index.mjs"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'
export const handler = async (event, context) => {
  const area = event.length * event.width;
  console.log(`The area is ${area}`);
  console.log('CloudWatch log group: ', context.logGroupName);
  return JSON.stringify({area});
};
NODEJS_EOF
        echo "Selected runtime: Node.js 22.x"
        ;;
```

```
    *)
        echo "ERROR: Invalid choice. Please enter 1 or 2."
        exit 1
        ;;
esac

# ============================================================================
# Step 1: Create IAM execution role
# ============================================================================

echo ""
echo "=========================================="
echo "Step 1: Create IAM execution role"
echo "=========================================="
echo ""

TRUST_POLICY='{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)
echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
```

```
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

################################################################################
# Step 2: Create Lambda function
################################################################################

echo ""
echo "========================================="
echo "Step 2: Create Lambda function"
echo "========================================="
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR"
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1
cd "$ORIGINAL_DIR"

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)
echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
```

```
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
query State --output text" \
    "Active"

###############################################################################
# Step 3: Invoke the function
###############################################################################

echo ""
echo "=========================================="
echo "Step 3: Invoke the function"
echo "=========================================="
echo ""

TEST_EVENT='{"length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

RESPONSE=$(cat "${TEMP_DIR}/response.json")
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

###############################################################################
# Step 4: View CloudWatch logs
###############################################################################

echo ""
echo "=========================================="
echo "Step 4: View CloudWatch Logs"
echo "=========================================="
```

```
echo ""

echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

################################################################################
# Summary and cleanup
################################################################################
```

```
echo ""
echo "========================================="
echo "SUMMARY"
echo "========================================="
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:   ${FUNCTION_NAME}"
echo "  CloudWatch logs:   ${LOG_GROUP_NAME}"
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    cleanup_resources
else
    echo ""
    echo "Resources were NOT deleted. To clean up manually, run:"
    echo ""
    echo "  # Delete the Lambda function"
    echo "  aws lambda delete-function --function-name ${FUNCTION_NAME}"
    echo ""
    echo "  # Delete the CloudWatch log group"
    echo "  aws logs delete-log-group --log-group-name ${LOG_GROUP_NAME}"
    echo ""
    echo "  # Detach the policy and delete the IAM role"
    echo "  aws iam detach-role-policy --role-name ${ROLE_NAME} --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole"
    echo "  aws iam delete-role --role-name ${ROLE_NAME}"
    echo ""

    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
fi

echo ""
echo "Done."
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteFunction
  - DeleteLogGroup
  - DeleteRole
  - DescribeLogStreams
  - DetachRolePolicy
  - GetFunction
  - GetFunctionConfiguration
  - GetLogEvents
  - Invoke

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 CloudWatch Logs 執行大型查詢

下列程式碼範例示範如何使用 CloudWatch Logs 查詢超過 10,000 筆記錄。

.NET

適用於 .NET 的 SDK (v4)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

這是示範大型查詢案例的主要工作流程。

```
using System.Diagnostics;
```

```
using System.Text.RegularExpressions;
using Amazon.CloudFormation;
using Amazon.CloudFormation.Model;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
using CloudWatchLogsActions;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;

namespace CloudWatchLogsScenario;

public class LargeQueryWorkflow
{
    /*
    Before running this .NET code example, set up your development environment,
 including your credentials.
    This .NET code example performs the following tasks for the CloudWatch Logs
 Large Query workflow:

    1. Prepare the Application:
       - Prompt the user to deploy CloudFormation stack and generate sample logs.
       - Deploy the CloudFormation template for resource creation.
       - Generate 50,000 sample log entries using CloudWatch Logs API.
       - Wait 5 minutes for logs to be fully ingested.

    2. Execute Large Query:
       - Perform recursive queries to retrieve all logs using binary search.
       - Display progress for each query executed.
       - Show total execution time and logs found.

    3. Clean up:
       - Prompt the user to delete the CloudFormation stack and all resources.
       - Destroy the CloudFormation stack and wait until removed.
    */

    public static ILogger<LargeQueryWorkflow> _logger = null!;
    public static CloudWatchLogsWrapper _wrapper = null!;
    public static IAmazonCloudFormation _amazonCloudFormation = null!;

    private static string _logGroupName = "/workflows/cloudwatch-logs/large-
query";
    private static string _logStreamName = "stream1";
    private static long _queryStartDate;
```

```
    private static long _queryEndDate;

    public static bool _interactive = true;
    public static string _stackName = "CloudWatchLargeQueryStack";
    private static string _stackResourcePath = "../../../../../../../scenarios/
features/cloudwatch_logs_large_query/resources/stack.yaml";

    public static async Task Main(string[] args)
    {
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter("Microsoft", LogLevel.Information))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonCloudWatchLogs>()
                    .AddAWSService<IAmazonCloudFormation>()
                    .AddTransient<CloudWatchLogsWrapper>()
            )
            .Build();

        if (_interactive)
        {
            _logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
                .CreateLogger<LargeQueryWorkflow>();

            _wrapper = host.Services.GetRequiredService<CloudWatchLogsWrapper>();
            _amazonCloudFormation =
 host.Services.GetRequiredService<IAmazonCloudFormation>();
        }

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the CloudWatch Logs Large Query
 Scenario.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("This scenario demonstrates how to perform large-scale
 queries on");
        Console.WriteLine("CloudWatch Logs using recursive binary search to
 retrieve more than");
        Console.WriteLine("the 10,000 result limit.");
        Console.WriteLine();

        try
        {
            Console.WriteLine(new string('-', 80));
```

```
            var prepareSuccess = await PrepareApplication();
            Console.WriteLine(new string('-', 80));

            if (prepareSuccess)
            {
                Console.WriteLine(new string('-', 80));
                await ExecuteLargeQuery();
                Console.WriteLine(new string('-', 80));
            }

            Console.WriteLine(new string('-', 80));
            await Cleanup();
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "There was a problem with the scenario,
initiating cleanup...");
            _interactive = false;
            await Cleanup();
        }

        Console.WriteLine("CloudWatch Logs Large Query scenario completed.");
    }

    /// <summary>
    /// Runs the scenario workflow. Used for testing.
    /// </summary>
    public static async Task RunScenario()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the CloudWatch Logs Large Query
Scenario.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("This scenario demonstrates how to perform large-scale
queries on");
        Console.WriteLine("CloudWatch Logs using recursive binary search to
retrieve more than");
        Console.WriteLine("the 10,000 result limit.");
        Console.WriteLine();

        try
        {
            Console.WriteLine(new string('-', 80));
```

```
            var prepareSuccess = await PrepareApplication();
            Console.WriteLine(new string('-', 80));

            if (prepareSuccess)
            {
                Console.WriteLine(new string('-', 80));
                await ExecuteLargeQuery();
                Console.WriteLine(new string('-', 80));
            }

            Console.WriteLine(new string('-', 80));
            await Cleanup();
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "There was a problem with the scenario,
initiating cleanup...");
            _interactive = false;
            await Cleanup();
        }

        Console.WriteLine("CloudWatch Logs Large Query scenario completed.");
    }

    /// <summary>
    /// Prepares the application by creating the necessary resources.
    /// </summary>
    /// <returns>True if the application was prepared successfully.</returns>
    public static async Task<bool> PrepareApplication()
    {
        Console.WriteLine("Preparing the application...");
        Console.WriteLine();

        try
        {
            var deployStack = !_interactive || GetYesNoResponse(
                "Would you like to deploy the CloudFormation stack and generate
sample logs? (y/n) ");

            if (deployStack)
            {
                if (_interactive)
                {
```

```
                    Console.Write(
                        $"Enter a path for the CloudFormation stack
resource .yaml file (or press Enter for default '{_stackResourcePath}'): ");
                    string? inputPath = Console.ReadLine();
                    if (!string.IsNullOrWhiteSpace(inputPath))
                    {
                        _stackResourcePath = inputPath;
                    }
                }

                _stackName = PromptUserForStackName();

                var deploySuccess = await DeployCloudFormationStack(_stackName);

                if (deploySuccess)
                {
                    Console.WriteLine();
                    Console.WriteLine("Generating 50,000 sample log entries...");
                    var generateSuccess = await GenerateSampleLogs();

                    if (generateSuccess)
                    {
                        Console.WriteLine();
                        Console.WriteLine("Sample logs created. Waiting 5 minutes
for logs to be fully ingested...");
                        await WaitWithCountdown(300);

                        Console.WriteLine("Application preparation complete.");
                        return true;
                    }
                }
            }
            else
            {
                _logGroupName = PromptUserForInput("Enter the log group name ",
_logGroupName);
                _logStreamName = PromptUserForInput("Enter the log stream name ",
_logStreamName);

                var startDateMs = PromptUserForLong("Enter the query start date
(milliseconds since epoch): ");
                var endDateMs = PromptUserForLong("Enter the query end date
(milliseconds since epoch): ");
```

```
                _queryStartDate = startDateMs / 1000;
                _queryEndDate = endDateMs / 1000;

                Console.WriteLine("Application preparation complete.");
                return true;
            }
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "An error occurred while preparing the
application.");
        }

        Console.WriteLine("Application preparation failed.");
        return false;
    }

    /// <summary>
    /// Deploys the CloudFormation stack with the necessary resources.
    /// </summary>
    /// <param name="stackName">The name of the CloudFormation stack.</param>
    /// <returns>True if the stack was deployed successfully.</returns>
    private static async Task<bool> DeployCloudFormationStack(string stackName)
    {
        Console.WriteLine($"\nDeploying CloudFormation stack: {stackName}");

        try
        {
            var request = new CreateStackRequest
            {
                StackName = stackName,
                TemplateBody = await File.ReadAllTextAsync(_stackResourcePath)
            };

            var response = await _amazonCloudFormation.CreateStackAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"CloudFormation stack creation started:
{stackName}");

                bool stackCreated = await
WaitForStackCompletion(response.StackId);
```

```
                if (stackCreated)
                {
                    Console.WriteLine("CloudFormation stack created
successfully.");
                    return true;
                }
                else
                {
                    _logger.LogError($"CloudFormation stack creation failed:
{stackName}");
                    return false;
                }
            }
            else
            {
                _logger.LogError($"Failed to create CloudFormation stack:
{stackName}");
                return false;
            }
        }
        catch (AlreadyExistsException)
        {
            _logger.LogWarning($"CloudFormation stack '{stackName}' already
exists. Please provide a unique name.");
            var newStackName = PromptUserForStackName();
            return await DeployCloudFormationStack(newStackName);
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while deploying the
CloudFormation stack: {stackName}");
            return false;
        }
    }

    /// <summary>
    /// Waits for the CloudFormation stack to be in the CREATE_COMPLETE state.
    /// </summary>
    /// <param name="stackId">The ID of the CloudFormation stack.</param>
    /// <returns>True if the stack was created successfully.</returns>
    private static async Task<bool> WaitForStackCompletion(string stackId)
    {
        int retryCount = 0;
        const int maxRetries = 30;
```

```
        const int retryDelay = 10000;

        while (retryCount < maxRetries)
        {
            var describeStacksRequest = new DescribeStacksRequest
            {
                StackName = stackId
            };

            var describeStacksResponse = await
_amazonCloudFormation.DescribeStacksAsync(describeStacksRequest);

            if (describeStacksResponse.Stacks.Count > 0)
            {
                if (describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.CREATE_COMPLETE)
                {
                    return true;
                }
                if (describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.CREATE_FAILED ||
                    describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.ROLLBACK_COMPLETE)
                {
                    return false;
                }
            }

            Console.WriteLine("Waiting for CloudFormation stack creation to
complete...");
            await Task.Delay(retryDelay);
            retryCount++;
        }

        _logger.LogError("Timed out waiting for CloudFormation stack creation to
complete.");
        return false;
    }

    /// <summary>
    /// Generates sample logs directly using CloudWatch Logs API.
    /// Creates 50,000 log entries spanning 5 minutes.
    /// </summary>
    /// <returns>True if logs were generated successfully.</returns>
```

```
    private static async Task<bool> GenerateSampleLogs()
    {
        const int totalEntries = 50000;
        const int entriesPerBatch = 10000;
        const int fiveMinutesMs = 5 * 60 * 1000;

        try
        {
            // Calculate timestamps
            var startTimeMs = DateTimeOffset.UtcNow.ToUnixTimeMilliseconds();
            var timestampIncrement = fiveMinutesMs / totalEntries;

            Console.WriteLine($"Generating {totalEntries} log entries...");

            var entryCount = 0;
            var currentTimestamp = startTimeMs;
            var numBatches = totalEntries / entriesPerBatch;

            // Generate and upload logs in batches
            for (int batchNum = 0; batchNum < numBatches; batchNum++)
            {
                var logEvents = new List<InputLogEvent>();

                for (int i = 0; i < entriesPerBatch; i++)
                {
                    logEvents.Add(new InputLogEvent
                    {
                        Timestamp =
 DateTimeOffset.FromUnixTimeMilliseconds(currentTimestamp).UtcDateTime,
                        Message = $"Entry {entryCount}"
                    });

                    entryCount++;
                    currentTimestamp += timestampIncrement;
                }

                // Upload batch
                var success = await _wrapper.PutLogEventsAsync(_logGroupName,
 _logStreamName, logEvents);
                if (!success)
                {
                    _logger.LogError($"Failed to upload batch {batchNum + 1}/
{numBatches}");
                    return false;
```

```
                }

                Console.WriteLine($"Uploaded batch {batchNum + 1}/{numBatches}");
            }

            // Set query date range (convert milliseconds to seconds for query
 API)
            _queryStartDate = startTimeMs / 1000;
            _queryEndDate = (currentTimestamp - timestampIncrement) / 1000;

            Console.WriteLine($"Query start date:
 {DateTimeOffset.FromUnixTimeSeconds(_queryStartDate):yyyy-MM-
ddTHH:mm:ss.fffZ}");
            Console.WriteLine($"Query end date:
 {DateTimeOffset.FromUnixTimeSeconds(_queryEndDate):yyyy-MM-ddTHH:mm:ss.fffZ}");
            Console.WriteLine($"Successfully uploaded {totalEntries} log
 entries");

            return true;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "An error occurred while generating sample
 logs.");
            return false;
        }
    }

    /// <summary>
    /// Executes the large query workflow.
    /// </summary>
    public static async Task ExecuteLargeQuery()
    {
        Console.WriteLine("Starting recursive query to retrieve all logs...");
        Console.WriteLine();

        var queryLimit = PromptUserForInteger("Enter the query limit (max 10000)
 ", 10000);
        if (queryLimit > 10000) queryLimit = 10000;

        var queryString = "fields @timestamp, @message | sort @timestamp asc";

        var stopwatch = Stopwatch.StartNew();
```

```
        var allResults = await PerformLargeQuery(_logGroupName, queryString,
_queryStartDate, _queryEndDate, queryLimit);
        stopwatch.Stop();

        Console.WriteLine();
        Console.WriteLine($"Queries finished in
{stopwatch.Elapsed.TotalSeconds:F3} seconds.");
        Console.WriteLine($"Total logs found: {allResults.Count}");

        // Check for duplicates
        Console.WriteLine();
        Console.WriteLine("Checking for duplicate logs...");
        var duplicates = FindDuplicateLogs(allResults);
        if (duplicates.Count > 0)
        {
            Console.WriteLine($"WARNING: Found {duplicates.Count} duplicate log
entries!");
            Console.WriteLine("Duplicate entries (showing first 10):");
            foreach (var dup in duplicates.Take(10))
            {
                Console.WriteLine($"  [{dup.Timestamp}] {dup.Message} (appears
{dup.Count} times)");
            }

            var uniqueCount = allResults.Count - duplicates.Sum(d => d.Count -
1);
            Console.WriteLine($"Unique logs: {uniqueCount}");
        }
        else
        {
            Console.WriteLine("No duplicates found. All logs are unique.");
        }
        Console.WriteLine();

        var viewSample = !_interactive || GetYesNoResponse("Would you like to see
a sample of the logs? (y/n) ");
        if (viewSample)
        {
            Console.WriteLine();
            Console.WriteLine($"Sample logs (first 10 of {allResults.Count}):");
            for (int i = 0; i < Math.Min(10, allResults.Count); i++)
            {
                var timestamp = allResults[i].Find(f => f.Field ==
"@timestamp")?.Value ?? "N/A";
```

```
                var message = allResults[i].Find(f => f.Field ==
 "@message")?.Value ?? "N/A";
                Console.WriteLine($"[{timestamp}] {message}");
            }
        }
    }

    /// <summary>
    /// Performs a large query using recursive binary search.
    /// </summary>
    private static async Task<List<List<ResultField>>> PerformLargeQuery(
        string logGroupName,
        string queryString,
        long startTime,
        long endTime,
        int limit)
    {
        var queryId = await _wrapper.StartQueryAsync(logGroupName, queryString,
 startTime, endTime, limit);
        if (queryId == null)
        {
            return new List<List<ResultField>>();
        }

        var results = await PollQueryResults(queryId);
        if (results == null || results.Count == 0)
        {
            return new List<List<ResultField>>();
        }

        var startDate =
 DateTimeOffset.FromUnixTimeSeconds(startTime).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
        var endDate = DateTimeOffset.FromUnixTimeSeconds(endTime).ToString("yyyy-
MM-ddTHH:mm:ss.fffZ");
        Console.WriteLine($"Query date range: {startDate} ({startTime}s) to
 {endDate} ({endTime}s). Found {results.Count} logs.");

        if (results.Count < limit)
        {
            Console.WriteLine($"  -> Returning {results.Count} logs (less than
 limit of {limit})");
            return results;
        }
```

```
        Console.WriteLine($"  -> Hit limit of {limit}. Need to split and
 recurse.");

        // Get the timestamp of the last log (sorted to find the actual last one)
        var lastLogTimestamp = GetLastLogTimestamp(results);
        if (lastLogTimestamp == null)
        {
            Console.WriteLine($"  -> No timestamp found in results. Returning
 {results.Count} logs.");
            return results;
        }

        Console.WriteLine($"  -> Last log timestamp: {lastLogTimestamp}");

        // Parse the timestamp and add 1 millisecond to avoid querying the same
 log again
        var lastLogDate = DateTimeOffset.Parse(lastLogTimestamp + " +0000");
        Console.WriteLine($"  -> Last log as DateTimeOffset: {lastLogDate:yyyy-
MM-ddTHH:mm:ss.fffZ} ({lastLogDate.ToUnixTimeSeconds()}s)");

        var offsetLastLogDate = lastLogDate.AddMilliseconds(1);
        Console.WriteLine($"  -> Offset timestamp (last
 + 1ms): {offsetLastLogDate:yyyy-MM-ddTHH:mm:ss.fffZ}
 ({offsetLastLogDate.ToUnixTimeSeconds()}s)");

        // Convert to seconds, but round UP to the next second to avoid
 overlapping with logs in the same second
        // This ensures we don't re-query logs that share the same second as the
 last log
        var offsetLastLogTime = offsetLastLogDate.ToUnixTimeSeconds();
        if (offsetLastLogDate.Millisecond > 0)
        {
            offsetLastLogTime++; // Move to the next full second
            Console.WriteLine($"  -> Adjusted to next full second:
 {offsetLastLogTime}s
 ({DateTimeOffset.FromUnixTimeSeconds(offsetLastLogTime):yyyy-MM-
ddTHH:mm:ss.fffZ})");
        }

        Console.WriteLine($"  -> Comparing:
 offsetLastLogTime={offsetLastLogTime}s vs endTime={endTime}s");
        Console.WriteLine($"  -> End time as date:
 {DateTimeOffset.FromUnixTimeSeconds(endTime):yyyy-MM-ddTHH:mm:ss.fffZ}");
```

```
        // Check if there's any time range left to query
        if (offsetLastLogTime >= endTime)
        {
            Console.WriteLine($"  -> No time range left to query. Offset time
 ({offsetLastLogTime}s) >= end time ({endTime}s)");
            return results;
        }

        // Split the remaining date range in half
        var (range1Start, range1End, range2Start, range2End) =
 SplitDateRange(offsetLastLogTime, endTime);

        var range1StartDate =
 DateTimeOffset.FromUnixTimeSeconds(range1Start).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
        var range1EndDate =
 DateTimeOffset.FromUnixTimeSeconds(range1End).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
        var range2StartDate =
 DateTimeOffset.FromUnixTimeSeconds(range2Start).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
        var range2EndDate =
 DateTimeOffset.FromUnixTimeSeconds(range2End).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");

        Console.WriteLine($"  -> Splitting remaining range:");
        Console.WriteLine($"     Range 1: {range1StartDate} ({range1Start}s) to
 {range1EndDate} ({range1End}s)");
        Console.WriteLine($"     Range 2: {range2StartDate} ({range2Start}s) to
 {range2EndDate} ({range2End}s)");

        // Query both halves recursively
        Console.WriteLine($"  -> Querying range 1...");
        var results1 = await PerformLargeQuery(logGroupName, queryString,
 range1Start, range1End, limit);
        Console.WriteLine($"  -> Range 1 returned {results1.Count} logs");

        Console.WriteLine($"  -> Querying range 2...");
        var results2 = await PerformLargeQuery(logGroupName, queryString,
 range2Start, range2End, limit);
        Console.WriteLine($"  -> Range 2 returned {results2.Count} logs");

        // Combine all results
```

```
        var allResults = new List<List<ResultField>>(results);
        allResults.AddRange(results1);
        allResults.AddRange(results2);

        Console.WriteLine($"  -> Combined total: {allResults.Count} logs
({results.Count} + {results1.Count} + {results2.Count})");

        return allResults;
    }

    /// <summary>
    /// Gets the timestamp string of the most recent log from a list of logs.
    /// Sorts timestamps to find the actual last one.
    /// </summary>
    private static string? GetLastLogTimestamp(List<List<ResultField>> logs)
    {
        var timestamps = logs
            .Select(log => log.Find(f => f.Field == "@timestamp")?.Value)
            .Where(t => !string.IsNullOrEmpty(t))
            .OrderBy(t => t)
            .ToList();

        if (timestamps.Count == 0)
        {
            return null;
        }

        return timestamps[timestamps.Count - 1];
    }

    /// <summary>
    /// Splits a date range in half.
    /// Range 2 starts at midpoint + 1 second to avoid overlap.
    /// </summary>
    private static (long range1Start, long range1End, long range2Start, long
range2End) SplitDateRange(long startTime, long endTime)
    {
        var midpoint = startTime + (endTime - startTime) / 2;
        // Range 2 starts at midpoint + 1 to avoid querying the same second twice
        return (startTime, midpoint, midpoint + 1, endTime);
    }

    /// <summary>
    /// Polls for query results until complete.
```

```
    /// </summary>
    private static async Task<List<List<ResultField>>?> PollQueryResults(string
queryId)
    {
        int retryCount = 0;
        const int maxRetries = 60;
        const int retryDelay = 1000;

        while (retryCount < maxRetries)
        {
            var response = await _wrapper.GetQueryResultsAsync(queryId);
            if (response == null)
            {
                return null;
            }

            if (response.Status == QueryStatus.Complete)
            {
                return response.Results;
            }

            if (response.Status == QueryStatus.Failed ||
                response.Status == QueryStatus.Cancelled ||
                response.Status == QueryStatus.Timeout ||
                response.Status == QueryStatus.Unknown)
            {
                _logger.LogError($"Query failed with status: {response.Status}");
                return null;
            }

            await Task.Delay(retryDelay);
            retryCount++;
        }

        _logger.LogError("Timed out waiting for query results.");
        return null;
    }

    /// <summary>
    /// Cleans up the resources created during the scenario.
    /// </summary>
    public static async Task<bool> Cleanup()
    {
        var cleanup = !_interactive || GetYesNoResponse(
```

```
            "Do you want to delete the CloudFormation stack and all resources?
(y/n) ");

        if (cleanup)
        {
            try
            {
                var stackDeleteSuccess = await
DeleteCloudFormationStack(_stackName, false);
                return stackDeleteSuccess;
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "An error occurred while cleaning up the
resources.");
                return false;
            }
        }

        Console.WriteLine($"Resources will remain. Stack name: {_stackName}, Log
group: {_logGroupName}");
        _logger.LogInformation("CloudWatch Logs Large Query scenario is
complete.");
        return true;
    }

    /// <summary>
    /// Deletes the CloudFormation stack and waits for confirmation.
    /// </summary>
    private static async Task<bool> DeleteCloudFormationStack(string stackName,
bool forceDelete)
    {
        var request = new DeleteStackRequest
        {
            StackName = stackName,
        };

        if (forceDelete)
        {
            request.DeletionMode = DeletionMode.FORCE_DELETE_STACK;
        }

        await _amazonCloudFormation.DeleteStackAsync(request);
```

```
        Console.WriteLine($"CloudFormation stack '{stackName}' is being deleted.
This may take a few minutes.");

        bool stackDeleted = await WaitForStackDeletion(stackName, forceDelete);

        if (stackDeleted)
        {
            Console.WriteLine($"CloudFormation stack '{stackName}' has been
deleted.");
            return true;
        }
        else
        {
            _logger.LogError($"Failed to delete CloudFormation stack
'{stackName}'.");
            return false;
        }
    }

    /// <summary>
    /// Waits for the stack to be deleted.
    /// </summary>
    private static async Task<bool> WaitForStackDeletion(string stackName, bool
forceDelete)
    {
        int retryCount = 0;
        const int maxRetries = 30;
        const int retryDelay = 10000;

        while (retryCount < maxRetries)
        {
            var describeStacksRequest = new DescribeStacksRequest
            {
                StackName = stackName
            };

            try
            {
                var describeStacksResponse = await
_amazonCloudFormation.DescribeStacksAsync(describeStacksRequest);

                if (describeStacksResponse.Stacks.Count == 0 ||
                    describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.DELETE_COMPLETE)
```

```
                {
                    return true;
                }

                if (!forceDelete && describeStacksResponse.Stacks[0].StackStatus
== StackStatus.DELETE_FAILED)
                {
                    return await DeleteCloudFormationStack(stackName, true);
                }
            }
            catch (AmazonCloudFormationException ex) when (ex.ErrorCode ==
"ValidationError")
            {
                return true;
            }

            Console.WriteLine($"Waiting for CloudFormation stack '{stackName}' to
be deleted...");
            await Task.Delay(retryDelay);
            retryCount++;
        }

        _logger.LogError($"Timed out waiting for CloudFormation stack
'{stackName}' to be deleted.");
        return false;
    }

    /// <summary>
    /// Waits with a countdown display.
    /// </summary>
    private static async Task WaitWithCountdown(int seconds)
    {
        for (int i = seconds; i > 0; i--)
        {
            Console.Write($"\rWaiting: {i} seconds remaining...  ");
            await Task.Delay(1000);
        }
        Console.WriteLine("\rWait complete.                      ");
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    private static bool GetYesNoResponse(string question)
```

```
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null && ynResponse.Equals("y",
StringComparison.InvariantCultureIgnoreCase);
        return response;
    }

    /// <summary>
    /// Prompts the user for a stack name.
    /// </summary>
    private static string PromptUserForStackName()
    {
        if (_interactive)
        {
            Console.Write($"Enter a name for the CloudFormation stack (press
Enter for default '{_stackName}'): ");
            string? input = Console.ReadLine();
            if (!string.IsNullOrWhiteSpace(input))
            {
                var regex = "[a-zA-Z][-a-zA-Z0-9]*";
                if (!Regex.IsMatch(input, regex))
                {
                    Console.WriteLine($"Invalid stack name. Using default:
{_stackName}");
                    return _stackName;
                }
                return input;
            }
        }
        return _stackName;
    }

    /// <summary>
    /// Prompts the user for input with a default value.
    /// </summary>
    private static string PromptUserForInput(string prompt, string defaultValue)
    {
        if (_interactive)
        {
            Console.Write($"{prompt}(press Enter for default '{defaultValue}'):
");
            string? input = Console.ReadLine();
            return string.IsNullOrWhiteSpace(input) ? defaultValue : input;
```

```
        }
        return defaultValue;
    }

    /// <summary>
    /// Prompts the user for an integer value.
    /// </summary>
    private static int PromptUserForInteger(string prompt, int defaultValue)
    {
        if (_interactive)
        {
            Console.Write($"{prompt}(press Enter for default '{defaultValue}'):
");
            string? input = Console.ReadLine();
            if (string.IsNullOrWhiteSpace(input) || !int.TryParse(input, out var
result))
            {
                return defaultValue;
            }
            return result;
        }
        return defaultValue;
    }

    /// <summary>
    /// Prompts the user for a long value.
    /// </summary>
    private static long PromptUserForLong(string prompt)
    {
        if (_interactive)
        {
            Console.Write(prompt);
            string? input = Console.ReadLine();
            if (long.TryParse(input, out var result))
            {
                return result;
            }
        }
        return 0;
    }

    /// <summary>
    /// Finds duplicate log entries based on timestamp and message.
    /// </summary>
```

```
    private static List<(string Timestamp, string Message, int Count)>
 FindDuplicateLogs(List<List<ResultField>> logs)
    {
        var logSignatures = new Dictionary<string, int>();

        foreach (var log in logs)
        {
            var timestamp = log.Find(f => f.Field == "@timestamp")?.Value ?? "";
            var message = log.Find(f => f.Field == "@message")?.Value ?? "";
            var signature = $"{timestamp}|{message}";

            if (logSignatures.ContainsKey(signature))
            {
                logSignatures[signature]++;
            }
            else
            {
                logSignatures[signature] = 1;
            }
        }

        return logSignatures
            .Where(kvp => kvp.Value > 1)
            .Select(kvp =>
            {
                var parts = kvp.Key.Split('|');
                return (Timestamp: parts[0], Message: parts[1], Count:
 kvp.Value);
            })
            .OrderByDescending(x => x.Count)
            .ToList();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
  - GetQueryResults
  - StartQuery

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

這是進入點。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { CloudWatchLogsClient } from "@aws-sdk/client-cloudwatch-logs";
import { CloudWatchQuery } from "./cloud-watch-query.js";

console.log("Starting a recursive query...");

if (!process.env.QUERY_START_DATE || !process.env.QUERY_END_DATE) {
  throw new Error(
    "QUERY_START_DATE and QUERY_END_DATE environment variables are required.",
  );
}

const cloudWatchQuery = new CloudWatchQuery(new CloudWatchLogsClient({}), {
  logGroupNames: ["/workflows/cloudwatch-logs/large-query"],
  dateRange: [
    new Date(Number.parseInt(process.env.QUERY_START_DATE)),
    new Date(Number.parseInt(process.env.QUERY_END_DATE)),
  ],
});

await cloudWatchQuery.run();

console.log(
  `Queries finished in ${cloudWatchQuery.secondsElapsed} seconds.\nTotal logs
 found: ${cloudWatchQuery.results.length}`,
);
```

這是一種類別，可在必要時將查詢拆分為多個步驟。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import {
  StartQueryCommand,
  GetQueryResultsCommand,
} from "@aws-sdk/client-cloudwatch-logs";
import { splitDateRange } from "@aws-doc-sdk-examples/lib/utils/util-date.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

class DateOutOfBoundsError extends Error {}

export class CloudWatchQuery {
  /**
   * Run a query for all CloudWatch Logs within a certain date range.
   * CloudWatch logs return a max of 10,000 results. This class
   * performs a binary search across all of the logs in the provided
   * date range if a query returns the maximum number of results.
   *
   * @param {import('@aws-sdk/client-cloudwatch-logs').CloudWatchLogsClient}
 client
   * @param {{ logGroupNames: string[], dateRange: [Date, Date], queryConfig:
 { limit: number } }} config
   */
  constructor(client, { logGroupNames, dateRange, queryConfig }) {
    this.client = client;
    /**
     * All log groups are queried.
     */
    this.logGroupNames = logGroupNames;

    /**
     * The inclusive date range that is queried.
     */
    this.dateRange = dateRange;

    /**
     * CloudWatch Logs never returns more than 10,000 logs.
     */
    this.limit = queryConfig?.limit ?? 10000;

    /**
     * @type {import("@aws-sdk/client-cloudwatch-logs").ResultField[][]}
     */
```

```
    this.results = [];
  }

  /**
   * Run the query.
   */
  async run() {
    this.secondsElapsed = 0;
    const start = new Date();
    this.results = await this._largeQuery(this.dateRange);
    const end = new Date();
    this.secondsElapsed = (end - start) / 1000;
    return this.results;
  }

  /**
   * Recursively query for logs.
   * @param {[Date, Date]} dateRange
   * @returns {Promise<import("@aws-sdk/client-cloudwatch-logs").ResultField[]
[]>}
   */
  async _largeQuery(dateRange) {
    const logs = await this._query(dateRange, this.limit);

    console.log(
      `Query date range: ${dateRange
        .map((d) => d.toISOString())
        .join(" to ")}. Found ${logs.length} logs.`,
    );

    if (logs.length < this.limit) {
      return logs;
    }

    const lastLogDate = this._getLastLogDate(logs);
    const offsetLastLogDate = new Date(lastLogDate);
    offsetLastLogDate.setMilliseconds(lastLogDate.getMilliseconds() + 1);
    const subDateRange = [offsetLastLogDate, dateRange[1]];
    const [r1, r2] = splitDateRange(subDateRange);
    const results = await Promise.all([
      this._largeQuery(r1),
      this._largeQuery(r2),
    ]);
    return [logs, ...results].flat();
```

```
}

/**
 * Find the most recent log in a list of logs.
 * @param {import("@aws-sdk/client-cloudwatch-logs").ResultField[][]} logs
 */
_getLastLogDate(logs) {
  const timestamps = logs
    .map(
      (log) =>
        log.find((fieldMeta) => fieldMeta.field === "@timestamp")?.value,
    )
    .filter((t) => !!t)
    .map((t) => `${t}Z`)
    .sort();

  if (!timestamps.length) {
    throw new Error("No timestamp found in logs.");
  }

  return new Date(timestamps[timestamps.length - 1]);
}

/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
  return this.client.send(new GetQueryResultsCommand({ queryId }));
}

/**
 * Starts a query and waits for it to complete.
 * @param {[Date, Date]} dateRange
 * @param {number} maxLogs
 */
async _query(dateRange, maxLogs) {
  try {
    const { queryId } = await this._startQuery(dateRange, maxLogs);
    const { results } = await this._waitUntilQueryDone(queryId);
    return results ?? [];
  } catch (err) {
    /**
     * This error is thrown when StartQuery returns an error indicating
```

```
       * that the query's start or end date occur before the log group was
       * created.
       */
      if (err instanceof DateOutOfBoundsError) {
        return [];
      }
      throw err;
    }
  }

  /**
   * Wrapper for the StartQueryCommand. Uses a static query string
   * for consistency.
   * @param {[Date, Date]} dateRange
   * @param {number} maxLogs
   * @returns {Promise<{ queryId: string }>}
   */
  async _startQuery([startDate, endDate], maxLogs = 10000) {
    try {
      return await this.client.send(
        new StartQueryCommand({
          logGroupNames: this.logGroupNames,
          queryString: "fields @timestamp, @message | sort @timestamp asc",
          startTime: startDate.valueOf(),
          endTime: endDate.valueOf(),
          limit: maxLogs,
        }),
      );
    } catch (err) {
      /** @type {string} */
      const message = err.message;
      if (message.startsWith("Query's end date and time")) {
        // This error indicates that the query's start or end date occur
        // before the log group was created.
        throw new DateOutOfBoundsError(message);
      }

      throw err;
    }
  }

  /**
   * Call GetQueryResultsCommand until the query is done.
   * @param {string} queryId
```

```
   */
  _waitUntilQueryDone(queryId) {
    const getResults = async () => {
      const results = await this._getQueryResults(queryId);
      const queryDone = [
        "Complete",
        "Failed",
        "Cancelled",
        "Timeout",
        "Unknown",
      ].includes(results.status);

      return { queryDone, results };
    };

    return retry(
      { intervalInMs: 1000, maxRetries: 60, quiet: true },
      async () => {
        const { queryDone, results } = await getResults();
        if (!queryDone) {
          throw new Error("Query not done.");
        }

        return results;
      },
    );
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
  - GetQueryResults
  - StartQuery

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此檔案會調用範例模組，以用於管理超過 10,000 筆結果的 CloudWatch 查詢。

```
import logging
import os
import sys

import boto3
from botocore.config import Config

from cloudwatch_query import CloudWatchQuery
from date_utilities import DateUtilities

# Configure logging at the module level.
logging.basicConfig(
    level=logging.INFO,
    format="%(asctime)s - %(levelname)s - %(filename)s:%(lineno)d - %(message)s",
)

DEFAULT_QUERY_LOG_GROUP = "/workflows/cloudwatch-logs/large-query"


class CloudWatchLogsQueryRunner:
    def __init__(self):
        """
        Initializes the CloudWatchLogsQueryRunner class by setting up date
 utilities
        and creating a CloudWatch Logs client with retry configuration.
        """
        self.date_utilities = DateUtilities()
        self.cloudwatch_logs_client = self.create_cloudwatch_logs_client()

    def create_cloudwatch_logs_client(self):
        """
```

```
        Creates and returns a CloudWatch Logs client with a specified retry
configuration.

        :return: A CloudWatch Logs client instance.
        :rtype: boto3.client
        """
        try:
            return boto3.client("logs", config=Config(retries={"max_attempts":
10}))
        except Exception as e:
            logging.error(f"Failed to create CloudWatch Logs client: {e}")
            sys.exit(1)

    def fetch_environment_variables(self):
        """
        Fetches and validates required environment variables for query start and
end dates.
        Fetches the environment variable for log group, returning the default
value if it
        does not exist.

        :return: Tuple of query start date and end date as integers and the log
group.
        :rtype: tuple
        :raises SystemExit: If required environment variables are missing or
invalid.
        """
        try:
            query_start_date = int(os.environ["QUERY_START_DATE"])
            query_end_date = int(os.environ["QUERY_END_DATE"])
        except KeyError:
            logging.error(
                "Both QUERY_START_DATE and QUERY_END_DATE environment variables
are required."
            )
            sys.exit(1)
        except ValueError as e:
            logging.error(f"Error parsing date environment variables: {e}")
            sys.exit(1)

        try:
            log_group = os.environ["QUERY_LOG_GROUP"]
        except KeyError:
```

```
            logging.warning("No QUERY_LOG_GROUP environment variable, using
 default value")
            log_group = DEFAULT_QUERY_LOG_GROUP

        return query_start_date, query_end_date, log_group

    def convert_dates_to_iso8601(self, start_date, end_date):
        """
        Converts UNIX timestamp dates to ISO 8601 format using DateUtilities.

        :param start_date: The start date in UNIX timestamp.
        :type start_date: int
        :param end_date: The end date in UNIX timestamp.
        :type end_date: int
        :return: Start and end dates in ISO 8601 format.
        :rtype: tuple
        """
        start_date_iso8601 =
 self.date_utilities.convert_unix_timestamp_to_iso8601(
            start_date
        )
        end_date_iso8601 = self.date_utilities.convert_unix_timestamp_to_iso8601(
            end_date
        )
        return start_date_iso8601, end_date_iso8601

    def execute_query(
        self,
        start_date_iso8601,
        end_date_iso8601,
        log_group="/workflows/cloudwatch-logs/large-query",
        query="fields @timestamp, @message | sort @timestamp asc"
    ):
        """
        Creates a CloudWatchQuery instance and executes the query with provided
 date range.

        :param start_date_iso8601: The start date in ISO 8601 format.
        :type start_date_iso8601: str
        :param end_date_iso8601: The end date in ISO 8601 format.
        :type end_date_iso8601: str
        :param log_group: Log group to search: "/workflows/cloudwatch-logs/large-
query"
        :type log_group: str
```

```
        :param query: Query string to pass to the CloudWatchQuery instance
        :type query: str
        """
        cloudwatch_query = CloudWatchQuery(
            log_group=log_group,
            query_string=query
        )
        cloudwatch_query.query_logs((start_date_iso8601, end_date_iso8601))
        logging.info("Query executed successfully.")
        logging.info(
            f"Queries completed in {cloudwatch_query.query_duration} seconds.
 Total logs found: {len(cloudwatch_query.query_results)}"
        )


def main():
    """
    Main function to start a recursive CloudWatch logs query.
    Fetches required environment variables, converts dates, and executes the
 query.
    """
    logging.info("Starting a recursive CloudWatch logs query...")
    runner = CloudWatchLogsQueryRunner()
    query_start_date, query_end_date, log_group =
 runner.fetch_environment_variables()
    start_date_iso8601 = DateUtilities.convert_unix_timestamp_to_iso8601(
        query_start_date
    )
    end_date_iso8601 =
 DateUtilities.convert_unix_timestamp_to_iso8601(query_end_date)
    runner.execute_query(start_date_iso8601, end_date_iso8601,
 log_group=log_group)


if __name__ == "__main__":
    main()
```

此模組會處理超過 10,000 筆結果的 CloudWatch 查詢。

```
import logging
import time
from datetime import datetime
```

```
import threading
import boto3

from date_utilities import DateUtilities

DEFAULT_QUERY = "fields @timestamp, @message | sort @timestamp asc"
DEFAULT_LOG_GROUP = "/workflows/cloudwatch-logs/large-query"

class DateOutOfBoundsError(Exception):
    """Exception raised when the date range for a query is out of bounds."""

    pass


class CloudWatchQuery:
    """
    A class to query AWS CloudWatch logs within a specified date range.

    :vartype date_range: tuple
    :ivar limit: Maximum number of log entries to return.
    :vartype limit: int
    :log_group str: Name of the log group to query
    :query_string str: query
    """

    def __init__(self, log_group: str = DEFAULT_LOG_GROUP, query_string:
 str=DEFAULT_QUERY) -> None:
        self.lock = threading.Lock()
        self.log_group = log_group
        self.query_string = query_string
        self.query_results = []
        self.query_duration = None
        self.datetime_format = "%Y-%m-%d %H:%M:%S.%f"
        self.date_utilities = DateUtilities()
        self.limit = 10000

    def query_logs(self, date_range):
        """
        Executes a CloudWatch logs query for a specified date range and
 calculates the execution time of the query.

        :return: A batch of logs retrieved from the CloudWatch logs query.
        :rtype: list
        """
```

```
        start_time = datetime.now()

        start_date, end_date = self.date_utilities.normalize_date_range_format(
            date_range, from_format="unix_timestamp", to_format="datetime"
        )

        logging.info(
            f"Original query:"
            f"\n       START:     {start_date}"
            f"\n       END:       {end_date}"
            f"\n       LOG GROUP: {self.log_group}"
        )
        self.recursive_query((start_date, end_date))
        end_time = datetime.now()
        self.query_duration = (end_time - start_time).total_seconds()

   def recursive_query(self, date_range):
        """
        Processes logs within a given date range, fetching batches of logs
recursively if necessary.

        :param date_range: The date range to fetch logs for, specified as a tuple
(start_timestamp, end_timestamp).
        :type date_range: tuple
        :return: None if the recursive fetching is continued or stops when the
final batch of logs is processed.
                 Although it doesn't explicitly return the query results, this
method accumulates all fetched logs
                 in the `self.query_results` attribute.
        :rtype: None
        """
        batch_of_logs = self.perform_query(date_range)
        # Add the batch to the accumulated logs
        with self.lock:
            self.query_results.extend(batch_of_logs)
        if len(batch_of_logs) == self.limit:
            logging.info(f"Fetched {self.limit}, checking for more...")
            most_recent_log = self.find_most_recent_log(batch_of_logs)
            most_recent_log_timestamp = next(
                item["value"]
                for item in most_recent_log
                if item["field"] == "@timestamp"
            )
            new_range = (most_recent_log_timestamp, date_range[1])
```

```
            midpoint = self.date_utilities.find_middle_time(new_range)

            first_half_thread = threading.Thread(
                target=self.recursive_query,
                args=((most_recent_log_timestamp, midpoint),),
            )
            second_half_thread = threading.Thread(
                target=self.recursive_query, args=((midpoint, date_range[1]),)
            )

            first_half_thread.start()
            second_half_thread.start()

            first_half_thread.join()
            second_half_thread.join()

    def find_most_recent_log(self, logs):
        """
        Search a list of log items and return most recent log entry.
        :param logs: A list of logs to analyze.
        :return: log
        :type :return List containing log item details
        """
        most_recent_log = None
        most_recent_date = "1970-01-01 00:00:00.000"

        for log in logs:
            for item in log:
                if item["field"] == "@timestamp":
                    logging.debug(f"Compared: {item['value']} to
{most_recent_date}")
                    if (
                        self.date_utilities.compare_dates(
                            item["value"], most_recent_date
                        )
                        == item["value"]
                    ):
                        logging.debug(f"New most recent: {item['value']}")
                        most_recent_date = item["value"]
                        most_recent_log = log
        logging.info(f"Most recent log date of batch: {most_recent_date}")
        return most_recent_log

    def perform_query(self, date_range):
```

```
        """
        Performs the actual CloudWatch log query.

        :param date_range: A tuple representing the start and end datetime for
the query.
        :type date_range: tuple
        :return: A list containing the query results.
        :rtype: list
        """
        client = boto3.client("logs")
        try:
            try:
                start_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
                )
                end_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
                )
                response = client.start_query(
                    logGroupName=self.log_group,
                    startTime=start_time,
                    endTime=end_time,
                    queryString=self.query_string,
                    limit=self.limit,
                )
                query_id = response["queryId"]
            except client.exceptions.ResourceNotFoundException as e:
                raise DateOutOfBoundsError(f"Resource not found: {e}")
            while True:
                time.sleep(1)
                results = client.get_query_results(queryId=query_id)
                if results["status"] in [
                    "Complete",
                    "Failed",
                    "Cancelled",
                    "Timeout",
                    "Unknown",
                ]:
                    return results.get("results", [])
        except DateOutOfBoundsError:
            return []
```

```
    def _initiate_query(self, client, date_range, max_logs):
        """
        Initiates the CloudWatch logs query.

        :param date_range: A tuple representing the start and end datetime for
the query.
        :type date_range: tuple
        :param max_logs: The maximum number of logs to retrieve.
        :type max_logs: int
        :return: The query ID as a string.
        :rtype: str
        """
        try:
            start_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
            )
            end_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
            )
            response = client.start_query(
                logGroupName=self.log_group,
                startTime=start_time,
                endTime=end_time,
                queryString=self.query_string,
                limit=max_logs,
            )
            return response["queryId"]
        except client.exceptions.ResourceNotFoundException as e:
            raise DateOutOfBoundsError(f"Resource not found: {e}")


    def _wait_for_query_results(self, client, query_id):
        """
        Waits for the query to complete and retrieves the results.

        :param query_id: The ID of the initiated query.
        :type query_id: str
        :return: A list containing the results of the query.
        :rtype: list
        """
        while True:
            time.sleep(1)
```

```
            results = client.get_query_results(queryId=query_id)
            if results["status"] in [
                "Complete",
                "Failed",
                "Cancelled",
                "Timeout",
                "Unknown",
            ]:
                return results.get("results", [])
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - GetQueryResults
  - StartQuery

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用排程事件來調用 Lambda 函式

下列程式碼範例示範如何建立由 Amazon EventBridge 排程事件呼叫的 AWS Lambda 函數。

Java

適用於 Java 2.x 的 SDK

顯示如何建立叫用 AWS Lambda 函數的 Amazon EventBridge 排程事件。將 EventBridge 設定為在調用 Lambda 函式時使用 Cron 表達式來進行排程。在此範例中，您會使用 Lambda Java 執行時期 API 建立 Lambda 函式。此範例會叫用不同的 AWS 服務來執行特定的使用案例。此範例示範如何建立應用程式，將行動裝置文字訊息傳送給員工，在他們的週年紀念日向他們道賀。

如需完整的原始碼和如何設定及執行的指示，請參閱 GitHub 上的完整範例。

此範例中使用的服務

- CloudWatch Logs
- DynamoDB

- EventBridge
- Lambda
- Amazon SNS

JavaScript

適用於 JavaScript (v3) 的 SDK

顯示如何建立叫用 AWS Lambda 函數的 Amazon EventBridge 排程事件。將 EventBridge 設定為在調用 Lambda 函式時使用 Cron 表達式來進行排程。在此範例中，您會使用 Lambda JavaScript 執行時期 API 建立 Lambda 函式。此範例會叫用不同的 AWS 服務來執行特定的使用案例。此範例示範如何建立應用程式，將行動裝置文字訊息傳送給員工，在他們的週年紀念日向他們道賀。

如需完整的原始碼和如何設定及執行的指示，請參閱 GitHub 上的完整範例。

此範例也可在 適用於 JavaScript 的 AWS SDK v3 開發人員指南中取得。

此範例中使用的服務

- CloudWatch Logs
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Python

適用於 Python 的 SDK (Boto3)

此範例說明如何將 AWS Lambda 函數註冊為排程 Amazon EventBridge 事件的目標。Lambda 處理常式會將合適的訊息和完整的事件資料寫入 Amazon CloudWatch Logs 中以供日後擷取。

- 部署 Lambda 函式。
- 建立一個 EventBridge 排程事件，並將 Lambda 函式做為目標。
- 授予許可讓 EventBridge 調用 Lambda 函式。
- 列印 CloudWatch Logs 中的最新資料，以顯示排程調用的結果。
- 清理示範期間建立的所有資源。

這個範例在 GitHub 上的檢視效果最佳。如需完整的原始碼和如何設定及執行的指示，請參閱 GitHub 上的完整範例。

此範例中使用的服務

- CloudWatch Logs
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用 CloudWatch Logs。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 加密 CloudWatch Logs 中的查詢資料表 AWS Key Management Service

查詢資料表資料一律會在 CloudWatch Logs 中加密。根據預設，CloudWatch Logs 會使用伺服器端加密搭配 256 位元進階加密標準 Galois/計數器模式 (AES-GCM) 來加密靜態查詢資料表資料。您也可以使用 AWS Key Management Service 進行此加密。如果您這麼做，則會使用 AWS KMS 金鑰完成加密。 AWS KMS 在查詢資料表層級啟用 加密，方法是在建立查詢資料表或更新查詢資料表時，將 KMS 金鑰與查詢資料表建立關聯。

> **Important**
>
> CloudWatch Logs 僅支援對稱 KMS 金鑰。請勿使用非對稱金鑰來加密查詢資料表中的資料。如需詳細資訊，請參閱使用對稱和非對稱金鑰。

將 KMS 金鑰與查詢資料表建立關聯後，所有存放在查詢資料表中的資料都會使用此金鑰加密。每當請求此資料時，CloudWatch Logs 會解密此資料。每當有人請求取得加密的資料時，CloudWatch Logs 必須擁有 KMS 金鑰許可。

如果您稍後取消 KMS 金鑰與查詢資料表的關聯，CloudWatch Logs 會使用 CloudWatch Logs 預設加密方法加密資料。不過，如果金鑰在您取消關聯之前已停用或刪除，則 CloudWatch Logs 無法讀取使用該金鑰加密的資料。

如需 CloudWatch Logs 如何使用 AWS KMS 加密日誌資料的一般資訊，請參閱 使用 在 CloudWatch Logs 中加密日誌資料 AWS Key Management Service。

CloudWatch Logs 如何使用 AWS KMS 查詢資料表

CloudWatch Logs 使用 AWS KMS 信封加密來保護查詢資料表資料。當您將 KMS 金鑰與查詢資料表建立關聯時，CloudWatch Logs 會將`GenerateDataKey`請求傳送至 AWS KMS。 AWS KMS 會產生唯一的資料加密金鑰 (DEK)，並同時傳回純文字複本和 DEK 的加密複本。CloudWatch Logs 使用純文字 DEK 來加密查詢資料表資料，然後將加密的 DEK 與加密的資料一起存放。不會儲存純文字 DEK，並在使用後從記憶體中捨棄。

當 CloudWatch Logs 需要讀取查詢資料表資料時，它會 AWS KMS 使用加密的 DEK 向 傳送`Decrypt`請求。 會 AWS KMS 解密 DEK，並將純文字 DEK 傳回 CloudWatch Logs，然後使用它來解密查詢資料表資料。

CloudWatch Logs 在向 提出請求時使用下列加密內容 AWS KMS：

```
{
    "aws:logs:arn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-name"
}
```

您可以在 IAM 政策和 AWS KMS 金鑰政策中使用此加密內容來控制對 KMS 金鑰的存取。如需詳細資訊，請參閱AWS KMS 金鑰和加密內容。

所需的 許可

若要搭配查詢資料表使用 AWS KMS 加密，IAM 主體必須具有 KMS 金鑰的下列 AWS KMS 許可：

- `kms:Decrypt`
- `kms:GenerateDataKey`

在以 KMS 金鑰加密的查詢資料表`GetLookupTable`上呼叫 時，需要 `kms:Decrypt`許可，以便 CloudWatch Logs 可以代表您解密資料。`StartQuery` 使用加密查詢表上使用 `lookup`命令的查詢呼叫 時，金鑰 （用於加密查詢表的 KMS 金鑰） 也需要 `kms:Decrypt` 許可。呼叫 `CreateLookupTable`或使用 `UpdateLookupTable` KMS 金鑰時需要 `kms:GenerateDataKey`許可，以便 CloudWatch Logs 可以產生資料加密金鑰來加密查詢資料表資料。

此外，CloudWatch Logs 服務必須具有使用 KMS 金鑰的許可。您可以將政策陳述式新增至 KMS 金鑰政策來授予這些許可，如下節所述。

步驟 1：建立 AWS KMS 金鑰

若要建立對稱 KMS 金鑰，請使用下列 create-key 命令：

```
aws kms create-key
```

輸出包含金鑰 ID 和金鑰的 Amazon Resource Name (ARN)。下列為範例輸出：

```
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1478910250.94,
        "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-
e40cb0d29f59",
        "AWSAccountId": "123456789012",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

記錄金鑰 ARN。您需要在下列步驟中使用它。

步驟 2：設定 KMS 金鑰許可

根據預設，所有 AWS KMS 金鑰都是私有的。只有資源擁有者可以使用它來加密和解密資料。您必須授予 CloudWatch Logs 服務主體使用金鑰的許可，並授予呼叫角色使用金鑰的許可。

首先，使用以下 get-key-policy 命令，將 KMS 金鑰的預設政策儲存為 `policy.json`：

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

在文字編輯器中開啟 `policy.json` 檔案，並新增下列陳述式，以授予 CloudWatch Logs 服務主體使用金鑰的許可。此範例使用符合加密內容的`Condition`區段，將金鑰限制為特定的查詢資料表。

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "logs.region.amazonaws.com"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
```

```
    "Condition": {
        "StringEquals": {
            "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:region:account-
id:lookup-table:lookup-table-name",
            "aws:SourceAccount": "account-id",
            "aws:SourceArn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-
name"
        }
    }
}
```

接著，將許可新增至將呼叫 CloudWatch Logs `CreateLookupTable`或 `UpdateLookupTable` APIs 的角色。CloudWatch Logs 會使用 代表客戶`kms:ViaService`呼叫 AWS KMS 。如需詳細資訊，請參閱 kms:ViaService。

```
{
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::account-id:role/role-name"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": [
                "logs.region.amazonaws.com"
            ]
        }
    }
}
```

最後，使用下列 put-key-policy 命令新增更新的政策：

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

步驟 3：將 KMS 金鑰與查詢資料表建立關聯

當您使用 CreateLookupTable API 建立 KMS 金鑰與查詢資料表建立關聯，或使用 UpdateLookupTable API 更新現有的查詢資料表。這兩個 APIs都是 AWSLogsConfigService 的一部分。

建立 KMS 金鑰時將其與查詢資料表建立關聯

使用 CreateLookupTable API，並使用 KMS 金鑰的 ARN 指定 kmsKeyArn 參數：

```
aws logs create-lookup-table \
    --lookup-table-name my-lookup-table \
    --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```

將 KMS 金鑰與現有查詢資料表建立關聯

使用 UpdateLookupTable API，並使用 KMS 金鑰的 ARN 指定 kmsKeyArn 參數：

```
aws logs update-lookup-table \
    --lookup-table-name my-lookup-table \
    --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```

考量事項

- 將 KMS 金鑰與查詢資料表建立關聯或取消關聯後，操作最多可能需要五分鐘才會生效。
- 如果您撤銷 CloudWatch Logs 對相關聯金鑰的存取權或刪除關聯的 KMS 金鑰，則無法再擷取 CloudWatch Logs 中的加密查詢資料表資料。
- 若要執行本主題中的步驟，您必須具有下列許可：kms:CreateKey、kms:PutKeyPolicy、kms:GetKeyPolicy和適當的 CloudWatch Logs 許可，才能呼叫 CreateLookupTable或 UpdateLookupTable。

Amazon CloudWatch Logs 中的安全

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。 AWS 也為您提供可安全使用的服務。在AWS 合規計畫中，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 WorkSpaces 的合規計劃，請參閱AWS 合規計劃的 服務範圍。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規

本文件有助於您了解如何在使用 Amazon CloudWatch Logs 時套用共同責任模型。說明如何設定 Amazon CloudWatch Logs 以符合您的安全性和合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 CloudWatch Logs 資源。

目錄

Amazon CloudWatch Logs 中的資料保護

> **Note**
>
> 除了下列有關 中一般資料保護的資訊之外 AWS，CloudWatch Logs 也可讓您遮罩日誌事件中的敏感資料。如需詳細資訊，請參閱使用遮罩功能協助保護敏感日誌資料。

AWS 共同責任模型適用於 Amazon CloudWatch Logs 中的資料保護。如此模型所述， AWS 負責保護執行所有 的 全球基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責

所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱AWS 安全性部落格上的AWS 共同責任模型和 GDPR 部落格文章。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 CloudWatch Logs 或使用主控台、API AWS CLI或 AWS SDKs的其他 AWS 服務 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

靜態加密

CloudWatch Logs 使用加密保護靜態資料。所有日誌群組都會經過加密。根據預設，CloudWatch Logs 服務會管理伺服器端加密，並使用伺服器端加密搭配 256 位元進階加密標準 Galois/計數器模式 (AES-GCM) 來加密靜態日誌資料。

如果您想要管理用於加密和解密日誌的金鑰，請使用 AWS KMS 金鑰。如需詳細資訊，請參閱使用 在 CloudWatch Logs 中加密日誌資料 AWS Key Management Service。

傳輸中加密

CloudWatch Logs 對傳輸中資料使用端對端加密。CloudWatch Logs 服務會管理伺服器端加密金鑰。

適用於 Amazon CloudWatch Logs 的 Identity and Access Management

存取 Amazon CloudWatch Logs 需要使用 AWS 登入資料來驗證您的請求。這些登入資料必須具有存取 AWS 資源的許可，例如擷取您雲端資源的 CloudWatch Logs 資料。以下章節提供詳細資訊，詳述如何使用 AWS Identity and Access Management (IAM) 與 CloudWatch Logs 來控制誰可存取，以協助保護資源的安全：

- 身分驗證
- 存取控制

身分驗證

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

存取控制

您可以持有效憑證來驗證請求，但還須具備許可，才能建立或存取 CloudWatch Logs 資源。例如，您必須有建立日誌串流、建立日誌群組等的許可。

以下章節說明如何管理 CloudWatch Logs 的許可。我們建議您先閱讀概觀。

- 管理 CloudWatch Logs 資源的存取許可概觀
- 對 CloudWatch Logs 使用以身分為基礎的政策 (IAM 政策)
- CloudWatch Logs 許可參考

管理 CloudWatch Logs 資源的存取許可概觀

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照《AWS IAM Identity Center 使用者指南》中的建立權限合集說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循《IAM 使用者指南》的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照《IAM 使用者指南》的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《IAM 使用者指南》的新增許可到使用者 (主控台) 中的指示。

主題

CloudWatch Logs 資源和操作

在 CloudWatch Logs 中，主資源為日誌群組、日誌串流和目的地。CloudWatch Logs 不支援子資源 (搭配主資源使用的其他資源)。

這些資源和子資源都有獨一無二的 Amazon Resource Name (ARN) 與其相關聯，如下表所示。

Resource Type (資源類型)	ARN 格式
日誌群組	以下兩種方式皆可使用。以 `:*` 結尾的第二種，是由 `describe-log-groups` CLI 命令和 DescribeLogGroups API 所傳回的。

Resource Type (資源類型)	ARN 格式
	arn:aws:logs:*region*:*account-id* :log-group:*log_group_name* arn:aws:logs:*region*:*account-id* :log-group:*log_group_name* :* 在下列情況下，使用第一個沒有結尾 :* 的版本： • 在許多 CloudWatch Logs APIs的 logGroupIdentifier 輸入欄位中。 • 在標記 APIs的 resourceArn 欄位中 • 在 IAM 政策中，指定 TagResource、UntagResource 和 ListTagsForResource 的許可時。 在 IAM 政策中指定所有其他 API 動作的許可時，使用第二個版本搭配結尾 來:*參考 ARN。
日誌串流	arn:aws:logs:*region*:*account-id* :log-group:*log_group_name* :log-stream:*log-stream-name*
目標	arn:aws:logs:*region*:*account-id* :destination:*destination_name*

如需 ARN 的詳細資訊，請參閱《IAM 使用者指南》中的 ARN。如需 CloudWatch Logs ARN 的相關資訊，請參閱 Amazon Web Services 一般參考 中的 Amazon Resource Name (ARN)。有關 CloudWatch Logs 適用的政策範例，請參閱對 CloudWatch Logs 使用以身分為基礎的政策 (IAM 政策)。

CloudWatch Logs 提供一組操作來使用 CloudWatch Logs 資源。如需可用操作的清單，請參閱 CloudWatch Logs 許可參考。

了解資源所有權

無論誰建立資源， AWS 帳戶都會擁有在帳戶中建立的資源。具體而言，資源擁有者是驗證資源建立請求的委託人實體 AWS 帳戶 （即根帳戶、使用者或 IAM 角色）。下列範例說明其如何運作：

- 如果您使用 AWS 帳戶的根帳戶登入資料來建立日誌群組， AWS 您的帳戶即為 CloudWatch Logs 資源的擁有者。
- 如果您在 AWS 帳戶中建立使用者，並將建立 CloudWatch Logs 資源的許可授予該使用者，則使用者可以建立 CloudWatch Logs 資源。不過，使用者所屬 AWS 的帳戶擁有 CloudWatch Logs 資源。
- 如果您在 AWS 帳戶中建立具有建立 CloudWatch Logs 資源許可的 IAM 角色，則任何可以擔任該角色的人都可以建立 CloudWatch Logs 資源。該角色所屬 AWS 的帳戶擁有 CloudWatch Logs 資源。

管理 資源的存取

許可政策描述誰可以存取哪些資源。下一節說明可用來建立許可政策的選項。

> **Note**
>
> 本節討論如何在 CloudWatch Logs 的環境中使用 IAM。它不提供 IAM 服務的詳細資訊。如需完整的 IAM 文件，請參閱《IAM 使用者指南》中的什麼是 IAM？。如需有關 IAM 政策語法和說明的資訊，請參閱《IAM 使用者指南》中的 IAM 政策參考。

連接到 IAM 身分的政策稱為身分類型政策 (IAM 政策)，而連接到資源的政策參考資源類型政策。CloudWatch Logs 支援以身分為基礎的政策，也支援以資源為基礎的政策來管理目的地，以用於啟用跨帳戶訂閱。如需詳細資訊，請參閱跨帳戶跨區域訂閱。

主題

- 日誌群組許可和 Contributor Insights
- 資源型政策

日誌群組許可和 Contributor Insights

Contributor Insights 是 CloudWatch 的一項功能，可讓您分析日誌群組的資料，以及建立時間序列來顯示參與者資料。您可以查看與前 N 個參與者有關的指標、唯一參與者的總數及其用量。如需詳細資訊，請參閱使用 Contributor Insights 來分析高基數資料。

當您將 `cloudwatch:PutInsightRule` 和 `cloudwatch:GetInsightRuleReport` 許可授予使用者時，該使用者就可以建立規則來評估 CloudWatch Logs 中的任何日誌群組，然後查看結果。結果可能包含這些日誌群組的參與者資料。請務必將這些許可只授予允許檢視此資料的使用者。

資源型政策

CloudWatch Logs 支援以資源為基礎的政策來管理目的地，可用於啟用跨帳戶訂閱。如需詳細資訊，請參閱步驟 1：建立目的地。您可以使用 PutDestination API 建立目的地，也可以使用 PutDestinationPolicy API 將資源政策新增至目的地。以下範例可讓帳戶 ID 為 111122223333 的另一個 AWS 帳戶將日誌群組訂閱到目的地 `arn:aws:logs:us-east-1:123456789012:destination:testDestination`。

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111122223333"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:us-
east-1:123456789012:destination:testDestination"
    }
  ]
}
```

指定政策元素：動作、效果和委託人

對於每項 CloudWatch Logs 資源，該服務定義一組 API 操作。CloudWatch Logs 定義一組您可在政策中指定的動作，以授予這些 API 操作的許可。為了執行 API 操作，某些 API 操作可能需要多個動作的

許可。如需資源與 API 操作的詳細資訊，請參閱 CloudWatch Logs 資源和操作 與 CloudWatch Logs 許可參考.

以下是基本的政策元素：

- 資源 - 您使用 Amazon Resource Name (ARN) 識別欲套用政策的資源。如需詳細資訊，請參閱CloudWatch Logs 資源和操作。
- 動作：使用動作關鍵字識別您要允許或拒絕的資源操作。例如，`logs.DescribeLogGroups` 許可允許使用者執行 `DescribeLogGroups` 操作。
- 效果 – 您可以指定使用者請求特定動作時會有什麼效果 (允許或拒絕)。如果您未明確授予存取 (允許) 資源，則隱含地拒絕存取。您也可以明確拒絕資源存取，這樣做可確保使用者無法存取資源，即使不同政策授予存取也是一樣。
- 委託人：在以身分為基礎的政策 (IAM 政策) 中，政策所連接的使用者就是隱含委託人。對於資源型政策，您可以指定想要收到許可的使用者、帳戶、服務或其他實體 (僅適用於資源型政策)。CloudWatch Logs 支援以資源為基礎的政策來管理目的地。

如需進一步了解 IAM 政策語法和說明，請參閱《IAM 使用者指南》中的 AWS IAM 政策參考。

如需查看所有 CloudWatch Logs API 動作及其適用資源的表格，請參閱 CloudWatch Logs 許可參考。

在政策中指定條件

當您授予許可時，可以使用存取政策語言來指定政策應該何時生效的條件。例如，建議只在特定日期之後套用政策。如需使用政策語言指定條件的詳細資訊，請參閱IAM 使用者指南中的條件。

欲表示條件，您可以使用預先定義的條件金鑰。如需每個 AWS 服務支援的內容金鑰清單，以及 AWS 全政策金鑰清單，請參閱AWS 服務和全域條件內容金鑰的動作、資源AWS 和條件金鑰。

> **Note**
>
> 您可以使用標籤來控制對 CloudWatch Logs 資源的存取，包括對日誌群組和目的地的存取。由於日誌群組與日誌串流之間的階層關係，系統會在日誌群組層級控制對日誌串流的存取。如需有關使用標籤來控制存取的詳細資訊，請參閱使用標籤控制對 Amazon Web Services 資源的存取。

對 CloudWatch Logs 使用以身分為基礎的政策 (IAM 政策)

這個主題提供以身分為基礎的政策範例，在該政策中帳戶管理員可以將許可政策連接至 IAM 身分 (即使用者、群組和角色)。

> **Important**
>
> 建議您先檢閱介紹主題，其中說明基本概念與選項，讓您管理對 CloudWatch Logs 資源的存取。如需詳細資訊，請參閱管理 CloudWatch Logs 資源的存取許可概觀。

本主題涵蓋下列項目：

- 使用 CloudWatch 主控台所需的許可
- AWS CloudWatch Logs 的受管（預先定義）政策
- 客戶管理政策範例

以下是許可政策的範例：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
    ],
      "Resource": [
        "arn:aws:logs:*:*:*"
    ]
  }
 ]
}
```

此政策有一個陳述式，將授予許可來建立日誌群組和日誌串流，以便將日誌事件上傳至日誌串流，以及列出日誌串流的詳細資訊。

`Resource` 值結尾的萬用字元 (*) 表示該陳述式允許對任何日誌群組執行 `logs:CreateLogGroup`、`logs:CreateLogStream`、`logs:PutLogEvents` 及 `logs:DescribeLogStreams` 動作的許可。若要限制此許可只提供給特定日誌群組，請將資源 ARN 中的萬用字元 (*) 更換為特定日誌群組 ARN。如需 IAM 政策陳述式中各區段的詳細資訊，請參閱《IAM 使用者指南》中的 IAM 政策元素參考。如需所有 CloudWatch Logs 動作的清單，請參閱 CloudWatch Logs 許可參考。

使用 CloudWatch 主控台所需的許可

若要讓使用者在 CloudWatch 主控台中使用 CloudWatch Logs，該使用者必須擁有一組最低許可，允許使用者描述其 AWS 帳戶中的其他 AWS 資源。為了在 CloudWatch 主控台使用 CloudWatch Logs，您必須有以下服務的許可：

- CloudWatch
- CloudWatch Logs
- OpenSearch Service
- IAM
- Kinesis
- Lambda
- Amazon S3

如果您建立比最基本必要許可更嚴格的 IAM 政策，則對於採取該 IAM 政策的使用者而言，主控台就無法如預期運作。為確保這些使用者仍可使用 CloudWatch 主控台，也請將 `CloudWatchReadOnlyAccess` 受管政策連接至使用者，如AWS CloudWatch Logs 的受管（預先定義）政策所述。

對於僅呼叫 AWS CLI 或 CloudWatch Logs API 的使用者，您不需要允許最低主控台許可。

對於不使用主控台來管理日誌訂閱的使用者，使用 CloudWatch 主控台所需的整套許可如下：

- cloudwatch:GetMetricData
- cloudwatch:ListMetrics
- logs:CancelExportTask
- logs:CreateExportTask

- logs:CreateLogGroup
- logs:CreateLogStream
- logs:DeleteLogGroup
- logs:DeleteLogStream
- logs:DeleteMetricFilter
- logs:DeleteQueryDefinition
- logs:DeleteRetentionPolicy
- logs:DeleteSubscriptionFilter
- logs:DescribeExportTasks
- logs:DescribeLogGroups
- logs:DescribeLogStreams
- logs:DescribeMetricFilters
- logs:DescribeQueryDefinitions
- logs:DescribeQueries
- logs:DescribeSubscriptionFilters
- logs:FilterLogEvents
- logs:GetLogEvents
- logs:GetLogGroupFields
- logs:GetLogRecord
- logs:GetQueryResults
- logs:PutMetricFilter
- logs:PutQueryDefinition
- logs:PutRetentionPolicy
- logs:StartQuery
- logs:StopQuery
- logs:PutSubscriptionFilter
- logs:TestMetricFilter

對於也將使用主控台來管理日誌訂閱的使用者而言，也需要以下許可：

- es:DescribeElasticsearchDomain

- es:ListDomainNames
- iam:AttachRolePolicy
- iam:CreateRole
- iam:GetPolicy
- iam:GetPolicyVersion
- iam:GetRole
- iam:ListAttachedRolePolicies
- iam:ListRoles
- kinesis:DescribeStreams
- kinesis:ListStreams
- lambda:AddPermission
- lambda:CreateFunction
- lambda:GetFunctionConfiguration
- lambda:ListAliases
- lambda:ListFunctions
- lambda:ListVersionsByFunction
- lambda:RemovePermission
- s3:ListBuckets

AWS CloudWatch Logs 的受管（預先定義）政策

AWS 透過提供由 建立和管理的獨立 IAM 政策，解決許多常見的使用案例 AWS。受管政策授與常見使用案例中必要的許可，讓您免於查詢需要哪些許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

下列 AWS 受管政策是 CloudWatch Logs 特有的，您可以連接到帳戶中的使用者和角色：

- CloudWatchLogsFullAccess – 授予對 CloudWatch Logs 的完整存取。
- CloudWatchLogsReadOnlyAccess – 授予對 CloudWatch Logs 的唯讀存取。

CloudWatchLogsFullAccess

CloudWatchLogsFullAccess 政策會授予對 CloudWatch Logs 的完整存取權。此政策包含 `cloudwatch:GenerateQuery`和 `cloudwatch:GenerateQueryResultsSummary`許可，因此使

用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。若要查看政策的完整內容，請參閱《 AWS 受管政策參考指南》中的 CloudWatchLogsFullAccess。

CloudWatchLogsReadOnlyAccess

CloudWatchLogsReadOnlyAccess 政策會授予對 CloudWatch Logs 的唯讀存取權。它包含 `cloudwatch:GenerateQuery`和 `cloudwatch:GenerateQueryResultsSummary`許可，因此使用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。若要查看政策的完整內容，請參閱《 AWS 受管政策參考指南》中的 CloudWatchLogsReadOnlyAccess。

CloudWatchOpenSearchDashboardsFullAccess

CloudWatchOpenSearchDashboardsFullAccess 政策授予許可，以建立、管理和刪除與 OpenSearch Service 的整合，以及在這些整合中建立刪除和管理自動產生的日誌儀表板。如需詳細資訊，請參閱使用 Amazon OpenSearch Service 進行分析。

若要查看政策的完整內容，請參閱《 AWS 受管政策參考指南》中的 CloudWatchOpenSearchDashboardsFullAccess。

CloudWatchOpenSearchDashboardAccess

CloudWatchOpenSearchDashboardAccess 政策授予檢視使用 Amazon OpenSearch Service 分析建立之已結束日誌儀表板的存取權。如需詳細資訊，請參閱使用 Amazon OpenSearch Service 進行分析。

> **Important**
>
> 除了授予此政策之外，若要讓角色或使用者能夠檢視付費日誌儀表板，您還必須在建立與 OpenSearch Service 的整合時指定它們。如需詳細資訊，請參閱步驟 1：建立與 OpenSearch Service 的整合。

若要查看政策的完整內容，請參閱《 AWS 受管政策參考指南》中的 CloudWatchOpenSearchDashboardAccess。

CloudWatchLogsCrossAccountSharingConfiguration

CloudWatchLogsCrossAccountSharingConfiguration 政策授予可建立、管理和檢視 Observability Access Manager 連結的存取權，以便在帳戶之間共用 CloudWatch Logs 資源。如需詳細資訊，請參閱 CloudWatch 跨帳戶觀察功能。

若要查看政策的完整內容，請參閱《 AWS 受管政策參考指南》中的 CloudWatchLogsCrossAccountSharingConfiguration。

CloudWatchLogsAPIKeyAccess

CloudWatchLogsAPIKeyAccess 政策會啟用 CloudWatch Logs API 金鑰身分驗證和加密日誌擷取。此政策授予許可，以使用承載權杖進行身分驗證，並將日誌事件寫入 CloudWatch Logs，並在日誌加密時提供其他解密和產生資料金鑰的 AWS KMS 許可。

此政策可授予下列許可：

- `logs` – 允許主體透過 API 金鑰承載字符進行驗證，並將日誌事件寫入 CloudWatch Logs 串流。
- `kms` – 允許主體讀取 AWS KMS 金鑰中繼資料、產生用於加密的資料金鑰，以及解密資料。這些許可允許服務使用客戶受管 AWS KMS 金鑰加密日誌資料，以支援加密的 CloudWatch Logs。存取僅限於透過 CloudWatch Logs 服務呼叫的操作。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 CloudWatchLogsAPIKeyAccess。

AWS 受管政策的 CloudWatch Logs 更新

檢視自此服務開始追蹤這些變更以來CloudWatch Logs AWS 受管政策更新的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 CloudWatch Logs 文件歷史記錄頁面上的 RSS 摘要。

變更	描述	Date
CloudWatchLogsAPIKeyAccess – 新政策。	CloudWatch Logs 新增了新的受管政策 CloudWatchLogsAPIKeyAccess。 此政策會啟用 CloudWatch Logs API 金鑰身分驗證和加密日誌擷取，授予使用承載字符進行身分驗證的許可，並將日誌事件寫入 CloudWatch Logs。	2026 年 2 月 17 日

變更	描述	Date
CloudWatchLogsFullAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsFullAccess。 新增可觀測性管理動作的許可，以允許唯讀存取遙測管道和 S3 資料表整合。	2025 年 12 月 2 日
CloudWatchLogsReadOnlyAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsReadOnlyAccess。 新增可觀測性管理動作的許可，以允許唯讀存取遙測管道和 S3 資料表整合。	2025 年 12 月 2 日
CloudWatchLogsFullAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsFullAccess。 `cloudwatch:GenerateQueryResultsSummary` 已新增 的許可，以允許產生查詢結果的自然語言摘要。	2025 年 5 月 20 日
CloudWatchLogsReadOnlyAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsReadOnlyAccess。 `cloudwatch:GenerateQueryResultsSummary` 已新增 的許可，以允許產生查詢結果的自然語言摘要。	2025 年 5 月 20 日

變更	描述	Date
CloudWatchLogsFullAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsFullAccess。 已新增 Amazon OpenSearch Service 和 IAM 的許可，以針對某些功能啟用 CloudWatch Logs 與 OpenSearch Service 的整合。	2024 年 12 月 1 日
CloudWatchOpenSearchDashboardsFullAccess – 新的 IAM 政策。	CloudWatch Logs 新增了新的 IAM 政策 CloudWatchOpenSearchDashboardsFullAccess。- 此政策授予建立、管理和刪除與 OpenSearch Service 整合的存取權，以及建立、管理和刪除這些整合中已結束的日誌儀表板。如需詳細資訊，請參閱使用 Amazon OpenSearch Service 進行分析。	2024 年 12 月 1 日
CloudWatchOpenSearchDashboardAccess – 新的 IAM 政策。	CloudWatch Logs 新增了新的 IAM 政策 CloudWatchOpenSearchDashboardAccess。- 此政策授予檢視由 提供之付費日誌儀表板的存取權 Amazon OpenSearch Service。如需詳細資訊，請參閱使用 Amazon OpenSearch Service 進行分析。	2024 年 12 月 1 日

變更	描述	Date
CloudWatchLogsFullAccess – 更新至現有政策。	CloudWatch Logs 已將許可新增至 CloudWatchLogsFullAccess。 已新增 `cloudwatch:GenerateQuery` 許可，因此使用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。	2023 年 11 月 27 日
CloudWatchLogsReadOnlyAccess – 更新至現有政策。	CloudWatch 已將許可新增至 CloudWatchLogsReadOnlyAccess。 已新增 `cloudwatch:GenerateQuery` 許可，因此使用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。	2023 年 11 月 27 日
CloudWatchLogsReadOnlyAccess – 更新為現有政策	CloudWatch Logs 已將許可新增至 CloudWatchLogsReadOnlyAccess。 已新增 `logs:StartLiveTail` 和 `logs:StopLiveTail` 許可，以便具有此政策的使用者可以使用主控台啟動和停止 CloudWatch Logs Live Tail 工作階段。如需詳細資訊，請參閱使用 Live Tail 以近乎即時的方式檢視日誌。	2023 年 6 月 6 日

變更	描述	Date
CloudWatchLogsCrossAccountSharingConfiguration – 新政策	CloudWatch Logs 新增了一項新政策，讓您能管理共用 CloudWatch Logs 日誌群組的 CloudWatch 跨帳戶觀察功能連結。 如需詳細資訊，請參閱 CloudWatch 跨帳戶觀察功能。	2022 年 11 月 27 日
CloudWatchLogsReadOnlyAccess – 更新為現有政策	CloudWatch Logs 已將許可新增至 CloudWatchLogsReadOnlyAccess。 新增 `oam:ListSinks` 和 `oam:ListAttachedLinks` 許可，以便採取此政策的使用者可以使用主控台，在 CloudWatch 跨帳戶觀察功能中檢視來源帳戶共用的資料。	2022 年 11 月 27 日

客戶管理政策範例

您可以建立自己的自訂 IAM 政策，以允許 CloudWatch Logs 動作與資源的許可。您可以將這些自訂政策連接至需要這些許可的使用者或群組。

在本節中，您可以找到使用者政策範例，以了解授予各種 CloudWatch Logs 動作的許可。當您使用 CloudWatch Logs API、 AWS 開發套件或 AWS CLI時，這些政策會起作用。

範例

- 範例 1：允許完整存取 CloudWatch Logs
- 範例 2：允許唯讀存取 CloudWatch Logs
- 範例 3：允許存取一個日誌群組

範例 1：允許完整存取 CloudWatch Logs

以下政策允許使用者存取所有 CloudWatch Logs 動作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

範例 2：允許唯讀存取 CloudWatch Logs

AWS 提供 CloudWatchLogsReadOnlyAccess 政策，可啟用 CloudWatch Logs 資料的唯讀存取。此政策包含以下許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:Describe*",
                "logs:Get*",
                "logs:List*",
                "logs:StartQuery",
                "logs:StopQuery",
                "logs:TestMetricFilter",
                "logs:FilterLogEvents",
                "logs:StartLiveTail",
                "logs:StopLiveTail",
```

```
                "cloudwatch:GenerateQuery"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

範例 3：允許存取一個日誌群組

以下政策允許使用者在一個指定的日誌群組中讀取和寫入日誌事件。

> **Important**
>
> 在 Resource 行中，需要日誌群組名稱末尾的 :* 來表示該政策適用於此日誌群組中的所有日誌串流。如果省略 :*，將不會強制執行此政策。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
      "Action": [
        "logs:CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:PutLogEvents",
        "logs:GetLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:us-west-2:123456789012:log-
group:SampleLogGroupName:*"
      }
   ]
}
```

使用標記和 IAM 政策在日誌群組層級進行控制

您可以授與使用者存取特定日誌群組，同時防止他們存取其他日誌群組。若要這樣做，請標記日誌群組，並使用 IAM 政策來參考這些標籤。若要將標籤套用至日誌群組，您必須擁有 `logs:TagResource` 或 `logs:TagLogGroup` 許可。無論您是在建立日誌群組時將標籤指派給日誌群組，或稍後將標籤指派給日誌群組，此許可要求均適用。

如需有關標籤日誌群組的詳細資訊，請參閱 在 Amazon CloudWatch Logs 中標記日誌群組。

當您標記日誌群組時，您就可以授予 IAM 政策給使用者，以只允許存取包含特定標籤的日誌群組。例如，以下政策陳述式只會授予標籤鍵 `Team` 值為 `Green` 日誌群組的存取。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:*"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/Team": "Green"
                }
            }
        }
    ]
}
```

StopQuery 和 StopLiveTail API 操作在傳統意義上不會與 AWS 資源互動。其不會傳回任何資料，放置任何資料，或以任何方式修改資源。而只是針對指定的 Live Tail 工作階段或指定的 CloudWatch Logs Insights 查詢進行操作，這些並未歸類為資源。因此，當您在 IAM 政策中針對這些操作指定 `Resource` 欄位時，必須將 `Resource` 欄位的值設定為 `*`，如下列範例所示。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement":
        [ {
            "Effect": "Allow",
            "Action": [
                "logs:StopQuery",
                "logs:StopLiveTail"
            ],
            "Resource": "*"
            }
        ]
}
```

如需有關使用 IAM 政策陳述式的詳細資訊，請參閱《IAM 使用者指南》中的使用政策控制存取。

CloudWatch Logs 許可參考

當您在設定 存取控制 並撰寫可連接到 IAM 身分 (以身分為基礎的政策) 的許可政策時，可以使用下列資料表作為參考。下表列出每個 CloudWatch Logs API 操作和您可以授予許可執行的相對應動作。您可以在政策的 `Action` 欄位中指定動作。針對 `Resource` 欄位，您可以指定日誌群組或日誌串流的 ARN，或是指定 `*` 來代表所有 CloudWatch Logs 資源。

您可以在 CloudWatch Logs 政策中使用 AWS整體條件金鑰來表達條件。如需 AWS全系列金鑰的完整清單，請參閱《AWS IAM 使用者指南》中的全域和 IAM 條件內容金鑰。

> **Note**
>
> 若要指定動作，請使用 `logs:` 前綴，後面接著 API 操作名稱。例如：。`logs:CreateLogGroup`、`logs:CreateLogStream` 或 `logs:*` (適用於所有 CloudWatch Logs 動作)。

CloudWatch Logs API 操作及動作所需的許可

CloudWatch Logs API 操作	所需許可 (API 動作)
CancelExportTask	`logs:CancelExportTask` 取消待處理或執行匯出任務時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
CreateExportTask	`logs:CreateExportTask` 從日誌群組將資料匯出至 Simple Storage Service (Amazon S3) 儲存貯體時為必要。
CreateLogGroup	`logs:CreateLogGroup` 建立新日誌群組時為必要。
CreateLogStream	`logs:CreateLogStream` 在日誌群組中建立新日誌串流時為必要。
DeleteDestination	`logs:DeleteDestination` 刪除日誌目的地及停用其任何訂閱篩選條件時為必要。
DeleteLogGroup	`logs:DeleteLogGroup` 刪除日誌群組及任何相關的存檔日誌事件時為必要。
DeleteLogStream	`logs:DeleteLogStream` 刪除日誌串流及任何相關的存檔日誌事件時為必要。
DeleteMetricFilter	`logs:DeleteMetricFilter` 刪除與日誌群組相關聯的指標篩選條件時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
DeleteQueryDefinition	`logs:DeleteQueryDefinition` 刪除 CloudWatch Logs Insights 中儲存的查詢定義時為必要。
DeleteResourcePolicy	`logs:DeleteResourcePolicy` 刪除 CloudWatch Logs 資源政策時為必要。
DeleteRetentionPolicy	`logs:DeleteRetentionPolicy` 刪除日誌群組的保留政策時為必要。
DeleteSubscriptionFilter	`logs:DeleteSubscriptionFilter` 刪除與日誌群組相關聯的訂閱篩選條件時為必要。
DescribeDestinations	`logs:DescribeDestinations` 檢視與帳戶相關的所有目的地時為必要。
DescribeExportTasks	`logs:DescribeExportTasks` 檢視與帳戶相關的所有匯出任務時為必要。
DescribeLogGroups	`logs:DescribeLogGroups` 檢視與帳戶相關的所有日誌群組時為必要。
DescribeLogStreams	`logs:DescribeLogStreams` 檢視與日誌群組相關的所有日誌串流時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
DescribeMetricFilters	`logs:DescribeMetricFilters` 檢視與日誌群組相關的所有指標時為必要。
DescribeQueryDefinitions	`logs:DescribeQueryDefinitions` 查看 CloudWatch Logs Insights 中儲存的查詢定義清單時為必要。
DescribeQueries	`logs:DescribeQueries` 查看已排程、正在執行或最近已執行的 CloudWatch Logs Insights 查詢清單時為必要。
DescribeResourcePolicies	`logs:DescribeResourcePolicies` 檢視 CloudWatch Logs 資源政策清單時為必要。
DescribeSubscriptionFilters	`logs:DescribeSubscriptionFilters` 檢視與日誌群組相關聯的所有訂閱篩選條件時為必要。
FilterLogEvents	`logs:FilterLogEvents` 依據日誌群組篩選條件模式排序日誌事件時為必要。
GetLogEvents	`logs:GetLogEvents` 從日誌串流擷取日誌事件時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
GetLogGroupFields	`logs:GetLogGroupFields` 擷取日誌群組內日誌事件中包含的欄位清單時為必要。
GetLogRecord	`logs:GetLogRecord` 從單一日誌事件擷取詳細資訊時為必要。
GetLogObject	`logs:GetLogRecord` 擷取透過 PutOpenTelemetryLogs API 擷取的大量日誌事件內容時需要。
GetQueryResults	`logs:GetQueryResults` 擷取 CloudWatch Logs Insights 查詢的結果時為必要。
ListEntitiesForLogGroup (僅限 CloudWatch 主控台的許可)	`logs:ListEntitiesForLogGroup` 尋找與日誌群組相關聯的實體時需要。在 CloudWatch 主控台中探索相關日誌時需要。
ListLogGroupsForEntity (僅限 CloudWatch 主控台的許可)	`logs:ListLogGroupsForEntity` 尋找與實體相關聯的日誌群組時需要。在 CloudWatch 主控台中探索相關日誌時需要。
ListTagsLogGroup	`logs:ListTagsLogGroup` 列出與日誌群組相關的標籤時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
ListLogGroups	`logs:DescribeLogGroups` 檢視與帳戶相關的所有日誌群組時為必要。
PutDestination	`logs:PutDestination` 需要建立或更新目的地日誌串流 (例如 Kinesis 串流) 時為必要。
PutDestinationPolicy	`logs:PutDestinationPolicy` 建立或更新與現有日誌目的地相關的存取政策時為必要。
PutLogEvents	`logs:PutLogEvents` 將日誌事件批次上傳至日誌串流時為必要。
PutMetricFilter	`logs:PutMetricFilter` 建立或更新指標篩選條件並將其與日誌群組建立關聯時為必要。
PutQueryDefinition	`logs:PutQueryDefinition` 在 CloudWatch Logs Insights 中儲存查詢時需要，包括具有參數的已儲存查詢。
PutResourcePolicy	`logs:PutResourcePolicy` 建立 CloudWatch Logs 資源政策時為必要。

CloudWatch Logs API 操作	所需許可 (API 動作)
PutRetentionPolicy	`logs:PutRetentionPolicy` 設定將日誌事件保持 (保留) 在日誌群組中的天數時為必要。
PutSubscriptionFilter	`logs:PutSubscriptionFilter` 建立或更新訂閱篩選條件並將其與日誌群組建立關聯時為必要。
StartQuery	`logs:StartQuery` 開始 CloudWatch Logs Insights 查詢時為必要。若要使用參數執行已儲存的查詢，您也需要 `logs:DescribeQueryDefinitions` 。
StopQuery	`logs:StopQuery` 停止進行中的 CloudWatch Logs Insights 查詢時為必要。
TagLogGroup	`logs:TagLogGroup` 新增或更新日誌群組標籤時為必要。
TestMetricFilter	`logs:TestMetricFilter` 針對日誌事件訊息的取樣來測試篩選條件模式時為必要。

對 CloudWatch Logs 使用服務連結角色

Amazon CloudWatch Logs 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是一種獨特的 IAM 角色，直接連結至 CloudWatch Logs。服務連結角色由 CloudWatch Logs 預先定義，並包含該服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可以更有效率設定 CloudWatch Logs，因為您不需要手動新增必要的許可。CloudWatch Logs 定義其服務連結角色的許可，除非另有定義，否則只有 CloudWatch Logs 可以擔任這些角色。已定義的許可包括信任政策和許可政策。該許可政策無法連接至其他任何 IAM 實體。

關於支援服務連結角色的其他服務，如需相關資訊，請參閱與 IAM 搭配運作的AWS 服務。尋找服務連結角色欄中顯示 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

CloudWatch Logs 的服務連結角色許可

CloudWatch Logs 使用名為 AWSServiceRoleForLogDelivery 的服務連結角色。CloudWatch Logs 使用此服務連結角色將日誌直接寫入 Firehose。如需詳細資訊，請參閱從 AWS 服務啟用記錄。

AWSServiceRoleForLogDelivery 服務連結角色信任下列服務擔任角色：

- `logs.amazonaws.com`

角色許可政策允許 CloudWatch Logs 對指定的資源完成下列動作：

- 動作： `firehose:PutRecord` 和 在所有 Firehose 串流`firehose:PutRecordBatch`上，其標籤的`LogDeliveryEnabled`索引鍵值為 `True`。當您建立訂閱以將日誌交付至 Firehose 時，此標籤會自動附加至 Firehose 串流。

您必須設定許可來允許 IAM 實體建立、編輯或刪除服務連結角色。此實體可以是使用者、群組或角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 CloudWatch Logs 的服務連結角色

您不需要手動建立服務連結角色。當您設定要直接傳送到 AWS 管理主控台、 AWS CLI或 AWS API 中 Firehose 串流的日誌時，CloudWatch Logs 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您再次設定日誌以直接傳送到 Firehose 串流時，CloudWatch Logs 會再次為您建立服務連結角色。

編輯 CloudWatch Logs 的服務連結角色

當您建立 AWSServiceRoleForLogDelivery 或其他任何服務連結角色後，CloudWatch Logs 就不允許您編輯角色。因為各種實體可能會參考角色，所以您無法變更角色的名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 CloudWatch Logs 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

> **Note**
>
> 當您嘗試刪除資源時，如果 CloudWatch Logs 服務正在使用該角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

刪除由 AWSServiceRoleForLogDelivery 服務連結角色所使用的 CloudWatch Logs 資源

- 停止將日誌直接傳送至 Firehose 串流。

使用 IAM 手動刪除服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForLogDelivery服務連結角色。如需詳細資訊，請參閱刪除服務連結角色

CloudWatch Logs 服務連結角色支援的區域

CloudWatch Logs 支援在提供服務的所有 AWS 區域中使用服務連結角色。如需詳細資訊，請參閱 CloudWatch Logs 區域和端點。

CloudWatch Logs 更新 AWS 服務連結角色

檢視自 CloudWatch Logs 開始追蹤這些變更以來 AWS ，服務連結角色的更新詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 CloudWatch Logs 文件歷史記錄頁面上的 RSS 摘要。

變更	描述	Date
AWSServiceRoleForLogDelivery 服務連結角色政策 — 更新現有政策	CloudWatch Logs 已變更 IAM 政策中與 AWSServiceRoleForLogDelivery 服務連結角色相關聯的許可。變更如下： • `firehose:ResourceTag/LogDeliveryEnabled": "true"` 條件金鑰已變更為 `aws:ResourceTag/LogDeliveryEnabled": "true"` 。	2021 年 7 月 15 日
CloudWatch Logs 開始追蹤變更	CloudWatch Logs 開始追蹤其 AWS 受管政策的變更。	2021 年 6 月 10 日

Amazon CloudWatch Logs 合規驗證

若要了解 AWS 服務 是否在特定合規計劃範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱AWS 合規計劃。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載報告 in AWS Artifact

您使用 時的合規責任 AWS 服務 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。如需使用 時合規責任的詳細資訊 AWS 服務，請參閱 AWS 安全文件。

Amazon CloudWatch Logs 中的復原功能

AWS 全球基礎設施是以 AWS 區域和可用區域為基礎建置。區域提供多個分開且隔離的實際可用區域，並以低延遲、高輸送量和高度備援網路連線相互連結。透過可用區域，您可以設計與操作的應用

程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

Amazon CloudWatch Logs 中的基礎設施安全

Amazon CloudWatch Logs 是受管服務，受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 CloudWatch Logs。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

搭配介面 VPC 端點使用 CloudWatch Logs

如果您使用 Amazon Virtual Private Cloud (Amazon VPC) 託管 AWS 資源，您可以在 VPC 和 CloudWatch Logs 之間建立私有連線。您可以使用此連線將日誌傳送到 CloudWatch Logs，而不需要透過網際網路傳送。CloudWatch Logs 支援所有區域中的 IPv4 VPC 端點，並支援所有區域中的 IPv6 端點。

Amazon VPC 是一種 AWS 服務，可用來在您定義的虛擬網路中啟動 AWS 資源。您可利用 VPC 來控制您的網路設定，例如 IP 地址範圍、子網路、路由表和網路閘道。若要將 VPC 連接到 CloudWatch Logs，請為 CloudWatch Logs 定義界面 VPC 端點。這類端點可讓您將 VPC 連線到 AWS 服務。端點提供可靠、可擴展的連線來連接 CloudWatch Logs，無須使用網際網路閘道、網路位址轉譯 (NAT) 執行個體或 VPN 連接。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC。

介面 VPC 端點採用 AWS PrivateLink 技術，這項 AWS 技術可使用具有私有 IP 地址的彈性網路介面，在 AWS 服務之間進行私有通訊。如需詳細資訊，請參閱新增 – AWS PrivateLink for AWS Services。

下列步驟適用於 Amazon VPC 的使用者。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的入門。

可用性

CloudWatch Logs 目前支援所有區域中 AWS 的 VPC 端點，包括 AWS GovCloud (US) 區域。

為 CloudWatch Logs 建立 VPC 端點

若要搭配 VPC 開始使用 CloudWatch Logs，請為 CloudWatch Logs 建立界面 VPC 端點。服務選擇為 com.amazonaws.***Region***.logs。若要與 FIPS 端點連線，要選擇的服務為 `com.amazonaws.Region.logs-fips`。您不需要變更 CloudWatch Logs 的任何設定。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的建立界面端點。

有些 CloudWatch Logs APIs，例如 StartLiveTail 和 GetLogObject，託管於不同的端點和 VPC 端點：`stream-logs.Region.amazonaws.com`。若要為這些 APIs 建立介面 VPC 端點，要選擇的服務為 com.amazonaws.***Region***.stream-logs。若要與 FIPS 端點連線，要選擇的服務為 `com.amazonaws.Region.stream-logs-fips`。

測試 VPC 和 CloudWatch Logs 之間的連線

建立端點後，您可以測試連線。

測試 VPC 和 CloudWatch Logs 端點之間的連線

1. 連線至位於 VPC 中的 Amazon EC2 執行個體。如需連線的詳細資訊，請參閱 Amazon EC2 文件中的連線至 Linux 執行個體或連線至 Windows 執行個體。
2. 從執行個體中，使用 在其中一個現有日誌群組中 AWS CLI 建立日誌項目。

   首先，以日誌事件建立一個 JSON 檔案。時間戳記必須以從 1970 年 1 月 1 日 00:00:00 UTC 之後的毫秒數指定。

   ```
   [
     {
       "timestamp": 1533854071310,
       "message": "VPC Connection Test"
     }
   ]
   ```

   然後使用 `put-log-events` 命令來建立日誌項目：

```
aws logs put-log-events --log-group-name LogGroupName --log-stream-
name LogStreamName --log-events file://JSONFileName
```

如果回應命令包含 nextSequenceToken，則該命令已成功而且您的 VPC 端點是正常運作的。

控制對 CloudWatch Logs VPC 端點的存取權

當您建立或修改端點時，VPC 端點政策是您連接至端點的 IAM 資源政策。如果您未在建立端點時連接政策，我們會以預設政策連接以允許完整存取服務。端點政策不會覆寫或取代 IAM 政策或服務特定的政策。這個另行區分的政策會控制從端點到所指定之服務的存取。

端點政策必須以 JSON 格式撰寫。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 端點控制服務的存取。

以下是 CloudWatch Logs 的端點政策範例。此政策可讓透過 VPC 連線到 CloudWatch Logs 的使用者建立日誌串流，並將日誌傳送到 CloudWatch Logs，還會防止使用者執行其他 CloudWatch Logs 動作。

```
{
  "Statement": [
    {
      "Sid": "PutOnly",
      "Principal": "*",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

修改 CloudWatch Logs 的 VPC 端點政策

1. 在 https://console.aws.amazon.com/vpc/ 開啟 Amazon VPC 主控台。
2. 在導覽窗格中選擇 Endpoints (端點)。

3. 如果您尚未建立 CloudWatch Logs 的端點，請選擇 Create Endpoint (建立端點)。然後選取 com.amazonaws.***Region***.logs，然後選擇 Create endpoint (建立端點)。
4. 選取 com.amazonaws.***Region***.logs 端點，然後選擇螢幕下半部的 Policy (政策) 標籤。
5. 選擇 Edit Policy (編輯政策)，並對政策做出變更。

VPC 內容金鑰支援

CloudWatch Logs 支援 `aws:SourceVpc` 和 `aws:SourceVpce` 內容金鑰，以限制存取特定 VPC 或特定 VPC 端點。這些金鑰只有在使用者使用 VPC 端點時才會運作。如需詳細資訊，請參閱《IAM 使用者指南》中的可用於部分服務的金鑰。

在 中記錄 CloudWatch Logs API 和主控台操作 AWS CloudTrail

Amazon CloudWatch Logs 已與 整合AWS CloudTrail，這項服務可提供使用者、角色或 所採取動作的記錄 AWS 服務。CloudTrail 會將 CloudWatch Logs 的 API 呼叫擷取為事件。擷取的呼叫包括來自 CloudWatch Logs 主控台的呼叫，以及對 CloudWatch Logs API 操作的程式碼呼叫。您可以使用 CloudTrail 所收集的資訊，判斷對 CloudWatch Logs 提出的請求、提出請求的 IP 地址、提出請求的時間，以及其他詳細資訊。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是使用根使用者還是使用者憑證提出。
- 請求是否代表 IAM Identity Center 使用者提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務服務提出。

當您建立帳戶 AWS 帳戶 時CloudTrail 會在 中處於作用中狀態，而且您會自動存取 CloudTrail 事件歷史記錄。CloudTrail 事件歷史記錄為 AWS 區域中過去 90 天記錄的管理事件，提供可檢視、可搜尋、可下載且不可變的記錄。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」中的使用 CloudTrail 事件歷史記錄。檢視事件歷史記錄不會產生 CloudTrail 費用。

如需 AWS 帳戶 過去 90 天內持續記錄的事件，請建立線索或 CloudTrail Lake 事件資料存放區。

CloudTrail 追蹤

線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。使用 建立的所有線索 AWS 管理主控台 都是多區域。您可以使用 AWS CLI建立單一或多區域追蹤。建議您建立多區域追蹤，因為您擷取 AWS 區域 帳戶中所有 的活動。如果您建立單一區域追蹤，您只能檢視追蹤 AWS 區域中記錄的事件。如需追蹤的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的為您的 AWS 帳戶建立追蹤和為組織建立追蹤。

您可以透過建立追蹤，免費將持續管理事件的一個複本從 CloudTrail 傳遞至您的 Amazon S3 儲存貯體，但這樣做會產生 Amazon S3 儲存費用。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。如需 Amazon S3 定價的相關資訊，請參閱 Amazon S3 定價。

CloudTrail Lake 事件資料存放區

CloudTrail Lake 讓您能夠對事件執行 SQL 型查詢。CloudTrail Lake 會將分列式 JSON 格式的現有事件轉換為 Apache ORC 格式。ORC 是一種單欄式儲存格式，針對快速擷取資料進行了最佳化。系統會將事件彙總到事件資料存放區中，事件資料存放區是事件的不可變集合，其依據為您透過套用進階事件選取器選取的條件。套用於事件資料存放區的選取器控制哪些事件持續存在並可供您查詢。如需 CloudTrail Lake 的詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 AWS CloudTrail Lake。

CloudTrail Lake 事件資料存放區和查詢會產生費用。建立事件資料存放區時，您可以選擇要用於事件資料存放區的定價選項。此定價選項將決定擷取和儲存事件的成本，以及事件資料存放區的預設和最長保留期。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。

CloudWatch Logs 支援將下列動作當作事件，記錄在 CloudTrail 日誌檔案中：

- AssociateKmsKey
- CancelExportTask
- CreateDelivery
- CreateExportTask
- CreateLogAnomalyDetector
- CreateLogGroup
- CreateLogStream
- DeleteAccountPolicy
- DeleteDataProtectionPolicy
- DeleteDelivery
- DeleteDeliveryDestination
- DeleteDeliveryDestinationPolicy
- DeleteDeliverySource
- DeleteDestination
- DeleteIndexPolicy
- DeleteIntegration
- DeleteLogAnomalyDetector
- DeleteLogGroup
- DeleteLogStream

- DeleteMetricFilter
- DeleteQueryDefinition
- DeleteResourcePolicy
- DeleteRetentionPolicy
- DeleteSubscriptionFilter
- DeleteTransformer
- DescribeAccountPolicies
- DescribeConfigurationTemplates
- DescribeDeliveries
- DescribeDeliveryDestinations
- DescribeDeliverySources
- DescribeDestinations
- DescribeExportTasks
- DescribeFieldIndexes
- DescribeIndexPolicies
- DescribeLogGroups
- DescribeLogStreams
- DescribeMetricFilters
- DescribeQueries
- DescribeQueryDefinitions
- DescribeResourcePolicies
- DescribeSubscriptionFilters
- DisassociateKmsKey
- FilterLogEvents
- GetDataProtectionPolicy
- GetDelivery
- GetDeliveryDestination
- GetDeliveryDestinationPolicy
- GetDeliverySource
- GetIntegration

- GetLogAnomalyDetector
- GetLogEvents
- GetLogGroupFields
- GetLogRecord
- GetQueryResults
- GetTransformer
- ListAnomalies
- ListIntegrations
- ListLogAnomalyDetectors
- ListLogGroups
- ListLogGroupsForQuery
- ListTagsForResource
- ListTagsLogGroup
- PutAccountPolicy
- PutDataProtectionPolicy
- PutDeliveryDestination
- PutDeliveryDestinationPolicy
- PutDeliverySource
- PutDestination
- PutDestinationPolicy
- PutIndexPolicy
- PutIntegration
- PutMetricFilter
- PutQueryDefinition
- PutResourcePolicy
- PutRetentionPolicy
- PutSubscriptionFilter
- PutTransformer
- StartLiveTail
- StartQuery

- StopQuery
- TagResource
- TestMetricFilter
- TestTransformer
- UntagResource
- UpdateAnomaly
- UpdateDeliveryConfiguration
- UpdateLogAnomalyDetector

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是否使用根或 IAM 使用者憑證提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

CloudTrail 中的查詢產生資訊

同時支援針對查詢產生器主控台事件的 CloudTrail 記錄。CloudWatch Logs Insights 和 CloudWatch Metric Insights 目前支援查詢產生器。在這些 CloudTrail 事件中，`eventSource` 是 `monitoring.amazonaws.com`。

下列範例顯示 CloudTrail 日誌項目，示範 CloudWatch Logs Insights 中的 GenerateQuery 動作。

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:assumed-role/role_name",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::111222333444:role/Administrator",
                "accountId": "123456789012",
                "userName": "SAMPLE_NAME"
            },
            "attributes": {
                "creationDate": "2020-04-08T21:43:24Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2020-04-08T23:06:30Z",
    "eventSource": "monitoring.amazonaws.com",
    "eventName": "GenerateQuery",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "exampleUserAgent",
    "requestParameters": {
        "query_ask": "***",
        "query_type": "LogsInsights",
        "logs_insights": {
            "fields": "***",
            "log_group_names": ["yourloggroup"]
        },
        "include_description": true
    },
    "responseElements": null,
    "requestID": "2f56318c-cfbd-4b60-9d93-1234567890",
    "eventID": "52723fd9-4a54-478c-ac55-1234567890",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


了解 日誌檔案項目

追蹤是一種組態，可讓事件以日誌檔案的形式交付至您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌專案。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

以下日誌檔案項目顯示使用者呼叫 CloudWatch Logs CreateExportTask 動作。


```
{
        "eventVersion": "1.03",
        "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::123456789012:user/someuser",
            "accountId": "123456789012",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "someuser"
        },
        "eventTime": "2016-02-08T06:35:14Z",
        "eventSource": "logs.amazonaws.com",
        "eventName": "CreateExportTask",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "127.0.0.1",
        "userAgent": "aws-sdk-ruby2/2.0.0.rc4 ruby/1.9.3 x86_64-linux Seahorse/0.1.0",
        "requestParameters": {
            "destination": "yourdestination",
            "logGroupName": "yourloggroup",
            "to": 123456789012,
            "from": 0,
            "taskName": "yourtask"
        },
        "responseElements": {
            "taskId": "15e5e534-9548-44ab-a221-64d9d2b27b9b"
        },
        "requestID": "1cd74c1c-ce2e-12e6-99a9-8dbb26bd06c9",
        "eventID": "fd072859-bd7c-4865-9e76-8e364e89307c",
        "eventType": "AwsApiCall",
        "apiVersion": "20140328",
        "recipientAccountId": "123456789012"
}
```

使用 CloudWatch 指標監控使用量

您可以使用本節中的資料表來檢閱 Amazon CloudWatch Logs 每分鐘傳送給 Amazon CloudWatch 的指標。

CloudWatch Logs 指標

AWS/Logs 命名空間包含下列指標。

指標	Description
CallCount	在您的帳戶中執行的特定 API 操作數目。 CallCount 是 CloudWatch Logs 服務用量指標。如需詳細資訊，請參閱 CloudWatch Logs 服務用量指標。 有效維度：Class, Resource, Service, Type 有效統計資訊：總和 單位：無
DeliveryErrors	將資料轉送至訂閱目的地時，CloudWatch Logs 接收到錯誤的日誌事件數量。如果目的地服務傳回可重試的錯誤，例如調節例外狀況或可重試的服務例外狀況 (例如 HTTP 5xx)，CloudWatch Logs 會繼續重試傳送，最多 24 小時。如果錯誤是不可重試的錯誤，例如 AccessDeniedException 或 ResourceNotFoundException ，則 CloudWatch Logs 不會嘗試重新傳送。 有效維度：LogGroupName、DestinationType、FilterName、PolicyLevel 有效統計資訊：總和 單位：無
DeliveryThrottling	將資料轉送至訂閱目的地時，CloudWatch Logs 遭調節的日誌事件數量。 如果目的地服務傳回可重試的錯誤，例如調節例外狀況或可重試的服務例外狀況 (例如 HTTP 5xx)，CloudWatch Logs 會繼續重試傳送，最多 24

指標	Description
	小時。如果錯誤是不可重試的錯誤，例如 AccessDeniedException 或 ResourceNotFoundException ，則 CloudWatch Logs 不會嘗試重新傳送。 有效維度：LogGroupName、DestinationType、FilterName、PolicyLevel 有效統計資訊：總和 單位：無
EMFDisabledErrors	由於符合日誌群組的 METRIC_EXTRACTION_POLICY 類型的作用中帳戶政策而忽略的 EMF 格式日誌事件數量。如需指標擷取政策的詳細資訊，請參閱 PutAccountPolicy。 有效維度：LogGroupName 有效統計資訊：總和 單位：無
EMFParsingErrors	處理內嵌指標格式日誌時遇到的剖析錯誤數量。如果日誌識別為內嵌指標格式，但未遵循正確格式，就會發生這類錯誤。如需有關內嵌指標格式的詳細資訊，請參閱規格：內嵌指標格式。 有效維度：LogGroupName 有效統計資訊：總和 單位：無

指標	Description
EMFValidationErrors	處理內嵌指標格式日誌時遇到的驗證錯誤數量。如果內嵌指標格式日誌中的指標定義不符合內嵌指標格式和 MetricDatum 規格，就會發生這類錯誤。如需有關 CloudWatch 內嵌指標格式的資訊，請參閱規格：內嵌指標格式。如需有關資料類型 MetricDatum 的資訊，請參閱《Amazon CloudWatch API 參考》中的 MetricDatum。 **Note** 某些驗證錯誤可能會導致 EMF 日誌中的多個指標無法發佈。例如，將捨棄具有無效命名空間的所有指標集。 有效維度：LogGroupName 有效統計資訊：總和 單位：無
ErrorCount	在您的帳戶中執行而導致錯誤的 API 操作數目。 ErrorCount 是 CloudWatch Logs 服務用量指標。如需詳細資訊，請參閱 CloudWatch Logs 服務用量指標。 有效維度：Class, Resource, Service, Type 有效統計資訊：總和 單位：無
ForwardedBytes	轉送至訂閱目的地的壓縮位元組中的日誌事件量。 有效維度：LogGroupName、DestinationType、FilterName 有效統計資訊：總和 單位：位元組

指標	Description
ForwardedLogEvents	轉送至訂閱目的地的日誌事件數量。 有效維度：LogGroupName、DestinationType、FilterName、PolicyLevel 有效統計資訊：總和 單位：無
IncomingBytes	上傳至 CloudWatch Logs 的未壓縮位元組中的日誌事件量。與 LogGroupName 維度搭配使用時，此為已上傳至日誌群組的未壓縮位元組中的日誌事件量。 有效維度：LogGroupName 有效統計資訊：總和 單位：位元組
IncomingLogEvents	上傳至 CloudWatch Logs 的日誌事件數量。與 LogGroupName 維度搭配使用時，此為已上傳至日誌群組的未壓縮位元組中的日誌事件數量。 有效維度：LogGroupName 有效統計資訊：總和 單位：無
LogEventsWithFindings	與使用 CloudWatch Logs 資料保護功能稽核的資料字串相符的日誌事件數。如需詳細資訊，請參閱使用遮罩功能協助保護敏感日誌資料。 有效維度：無 有效統計資訊：總和 單位：無

指標	Description
ThrottleCount	在您的帳戶中因用量配額而調節執行的 API 操作數目。 ThrottleCount 是 CloudWatch Logs 服務用量指標。如需詳細資訊，請參閱 CloudWatch Logs 服務用量指標。 有效維度：Class, Resource, Service, Type 有效統計資訊：總和 單位：無

CloudWatch Logs 指標的維度

您可以搭配大多數 CloudWatch Logs 指標使用的維度列於下表中。

維度	Description
LogGroupName	要顯示指標的 CloudWatch Logs 日誌群組的名稱。
DestinationType	CloudWatch Logs 資料的訂閱目的地，可以是 AWS Lambda、Amazon Kinesis Data Streams 或 Amazon Data Firehose。
FilterName	將資料從日誌群組轉送至目的地的訂閱篩選器名稱。訂閱篩選條件名稱由 CloudWatch 自動轉換為 ASCII，並以問號 (?) 取代任何不支援的字元。

訂閱篩選條件指標維度

下表列出與帳戶層級訂閱篩選條件相關的指標維度。

維度	Description
PolicyLevel	政策適用的層級。目前，此維度的唯一有效值是 AccountPolicy

維度	Description
DestinationType	CloudWatch Logs 資料的訂閱目的地，可以是 AWS Lambda、Amazon Kinesis Data Streams 或 Amazon Data Firehose。
FilterName	將資料從日誌群組轉送至目的地的訂閱篩選器名稱。訂閱篩選條件名稱由 CloudWatch 自動轉換為 ASCII，並以問號 (?) 取代任何不支援的字元。

日誌轉換器指標和維度

CloudWatch Logs 會將下列日誌轉換器指標發佈至 AWS/Logs 命名空間中的 CloudWatch。

指標	Description
TransformationErrors	使用指定的轉換器轉換日誌事件時遇到的錯誤數目。 單位：無 有效統計資訊：總和
TransformedBytes	轉換日誌事件輸出的磁碟區，以未壓縮的位元組為單位。 單位：位元組 有效統計資訊：總和
TransformedLogEvents	轉換日誌事件的數量。 單位：無 有效統計資訊：總和

轉換器指標會使用下列維度。

維度	Description
LogGroupname	此維度僅用於log-group-level轉換器。
PolicyLevel	此維度僅用於帳戶層級轉換器。目前此維度的唯一有效值是 AccountPolicy

集中化指標和維度

若要跨多個帳戶和區域進行集中監控，您可以使用 CloudWatch Logs Centralization 在中央位置合併日誌資料和指標。如需詳細資訊，請參閱跨帳戶跨區域日誌集中化。

CloudWatch Logs 會將下列集中化指標發佈至 AWS/Logs 命名空間中的 CloudWatch。這些指標可協助您在使用 CloudWatch Logs 集中化規則時，監控從來源帳戶到目的地帳戶的日誌資料複寫。

> **Note**
>
> CloudWatch 會在您建立集中化規則後立即開始報告集中化指標。指標會盡最大努力發佈，並僅追蹤建立集中化規則後抵達的新日誌事件。

指標	Description	發佈於
IncomingCopiedBytes	複寫至目的地帳戶的未壓縮位元組中的日誌資料量。此指標僅適用於建立集中化規則後抵達的新日誌事件。 有效維度：SourceRegion、SourceAccount 有效統計資訊：總和 單位：位元組	目的地帳戶
IncomingCopiedLogEvents	複寫至目的地帳戶的日誌事件數目。此指標僅適用於建立集中化規則後抵達的新日誌事件。 有效維度：SourceRegion、SourceAccount	目的地帳戶

指標	Description	發佈於
	有效統計資訊：總和 單位：無	
OutgoingCopiedBytes	從來源帳戶傳送至目的地帳戶的未壓縮位元組日誌資料量。此指標僅適用於建立集中化規則後複製到目的地的新日誌事件。 有效維度：DestinationRegion 有效統計資訊：總和 單位：位元組	來源帳戶
OutgoingCopiedLogEvents	從來源帳戶傳送至目的地帳戶的日誌事件數目。此指標僅適用於建立集中化規則後複製到目的地的新日誌事件。 有效維度：DestinationRegion 有效統計資訊：總和 單位：無	來源帳戶
CentralizationError	複寫日誌資料時遇到的錯誤數目。由於 KMS 金鑰許可問題、日誌群組配額限制或日誌層不相符等各種原因，可能會發生錯誤。若要識別複寫失敗的特定日誌群組或日誌串流及其失敗原因，請監控 ErrorType 維度。 有效維度：ErrorType 有效統計資訊：總和 單位：無	目的地帳戶

指標	Description	發佈於
CentralizationThrottled	集中處理受到調節的次數。調節會導致集中處理速度變慢，但無法防止複寫日誌資料。 有效維度：DestinationRegion 有效統計資訊：總和 單位：無	來源帳戶

集中化指標會使用下列維度。

維度	Description
SourceRegion	來源日誌資料的來源 AWS 區域。
SourceAccount	來源日誌資料來源 AWS 的帳戶 ID。
DestinationRegion	要複寫日誌資料 AWS 的區域。
ErrorType	在集中化期間遇到的錯誤類型。可能的值包括： • LogGroupQuotaExceeded ：目的地帳戶已達到其日誌群組配額。 • InvalidKMS ：KMS 金鑰遺失、刪除、未啟用，或為非對稱而非對稱。 • AccessDenied ：嘗試複寫日誌資料時，存取遭拒。這可能是由於許可不足，例如 KMS 金鑰許可不正確、缺少 IAM 許可或限制性資源政策。 • LogTierMismatch ：來源和目的地日誌群組的日誌層不相符。 • InvalidLogStream ：建立或寫入日誌串流時遇到無效的參數，例如當日誌串流名稱超過長度上限時。 • InvalidLogGroup ：在目的地帳戶中建立或設定日誌群組時遇到無效的參數。

維度	Description
	• DestinationEncryptionMismatch ：目的地日誌群組已存在，其 KMS 加密組態與集中規則指定的組態不同。

CloudWatch Logs 服務用量指標

CloudWatch Logs 會將指標傳送至 CloudWatch，以追蹤 CloudWatch Logs API 操作的使用情況。這些指標對應至 AWS 服務配額。追蹤這些指標可協助您主動管理配額。如需詳細資訊，請參閱 Service Quotas 整合與用量指標。

例如，您可以追蹤 ThrottleCount 指標，或在該指標設定警示。如果此指標的值上升，您應該考慮對遭調節的 API 操作要求增加配額。如需 CloudWatch Logs 服務配額的詳細資訊，請參閱 CloudWatch Logs 配額。

CloudWatch Logs 每分鐘會在 AWS/Usage 和 AWS/Logs 命名空間發佈服務配額用量指標。

下表列出 CloudWatch Logs 發佈的服務用量指標。這些指標沒有規定的單位。這些指標最實用的統計數字是 SUM，代表 1 分鐘期間的總操作計數。

每個指標會連同 Service、Class、Type 及Resource 全部維度的值一起發佈。也會連同稱為 Account Metrics 的單一維度一起發佈。使用 Account Metrics 維度查看帳戶中所有 API 操作的指標總和。使用其他維度，並指定 API 操作的名稱給 Resource 維度，以尋找該特定 API 的指標。

指標

指標	Description
CallCount	在您的帳戶中執行的指定操作數目。 CallCount 會發佈在 AWS/Usage 和 AWS/Logs 命名空間。
ErrorCount	在您的帳戶中執行而導致錯誤的 API 操作數目。 ErrorCount 只會發佈在 AWS/Logs。
ThrottleCount	在您的帳戶中因用量配額而調節執行的 API 操作數目。 ThrottleCount 只會發佈在 AWS/Logs。

Dimensions (尺寸)

維度	Description
`Account metrics`	使用此維度可取得所有 CloudWatch Logs API 中指標總和。 如果您想要查看某個特定 API 的指標，請使用此表格中列出的其他維度，並指定 API 名稱作為 `Resource` 的值。
`Service`	包含資源 AWS 的服務名稱。對於 CloudWatch Logs 用量指標，此維度的值為 `Logs`。
`Class`	正在追蹤的資源類別。CloudWatch Logs API 用量指標使用此維度搭配值 None。
`Type`	正在追蹤的資源類型。目前，當 `Service` 維度為 `Logs`，`Type` 的唯一有效值為 `API`。
`Resource`	API 操作的名稱。有效值包括所有列在動作中的 API 操作名稱。例如 `PutLogEvents`

CloudWatch Logs 配額

您可以使用本節中的表格來檢閱 Amazon CloudWatch Logs 中 AWS 帳戶的預設服務配額，也稱為限制。大多數服務配額，但不是全部，都列在 Service Quotas 主控台的 Amazon CloudWatch Logs 命名空間下。 Service Quotas

> **Note**
>
> 如果您想要請求提高任何這些配額，請參閱本節中的程序。

名稱	預設	可調整	Description
作用中匯出任務	每個受支援的區域：1	否	每個帳戶的作用中（執行中或待定）匯出任務數量
每秒交易的 AssociateKmsKey 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域每秒的 associate-kms-key 呼叫數量上限
每秒交易的 AssociateSourceToS3TableIntegration 節流限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 AssociateSourceToS3TableIntegration 呼叫數上限
AssociateSourceToS3TableIntegration 調節限制	每個受支援的區域：10	否	爆量中每個帳戶/每個區域的每秒 AssociateSourceToS3TableIntegration 呼叫數量上限
批次大小	每個受支援的區域：1 MB	否	put-log-events 請求的最大批次大小，以 MB 為單位

名稱	預設	可調整	Description
每秒交易的 CancelExportTask 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 cancel-export-task 呼叫數量上限
每秒交易的 CancelImportTask 限流限制	每個受支援的區域：1	否	每個帳戶/每個區域的每秒 CancelImportTask 呼叫數上限
高載中每秒交易的 CancelImportTask 限流限制	每個受支援的區域：1	否	高載中每個帳戶/每個區域的每秒 CancelImportTask 呼叫數量上限
每秒交易的 CreateExportTask 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 create-export-task 呼叫數量上限
每秒交易的 CreateImportTask 限流限制	每個受支援的區域：1	否	每個帳戶/每個區域的每秒 CreateImportTask 呼叫數目上限
高載中每秒交易的 CreateImportTask 限流限制	每個受支援的區域：1	否	高載中每個帳戶/每個區域的每秒 CreateImportTask 呼叫數量上限
每秒交易的 CreateLogGroup 調節限制	每個支援的區域：每秒 10	是	每個帳戶/每個區域的每秒 create-log-group 呼叫數量上限
每秒交易的 CreateLogStream 調節限制	每個支援的區域：每秒 50 個	是	每個帳戶/每個區域的每秒 create-log-stream 呼叫數量上限
資料存檔	所有受支援的區域：5 GB	否	可用資料封存的 GB 大小上限

名稱	預設	可調整	Description
每秒交易的 DeleteDataProtectionPolicy 調節限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 delete-data-protection-policy 呼叫數目上限
每秒交易的 DeleteDestination 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒刪除目的地呼叫數上限
每秒交易的 DeleteLogGroup 調節限制	每個支援的區域：每秒 10	是	每個帳戶/每個區域的每秒 delete-log-group 呼叫數量上限
每秒交易中的 DeleteLogStream 調節限制	每個支援的區域：每秒 15 個	是	每個帳戶/每個區域的每秒 delete-log-stream 呼叫數量上限
每秒交易的 DeleteMetricFilter 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 delete-metric-filter呼叫數目上限
每秒交易的 DeleteRetentionPolicy 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 delete-retention-policy 呼叫數上限
每秒交易的 DeleteSubscriptionFilter 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 delete-subscription-filter 呼叫數上限
每秒交易中的 DeleteTransformer 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒刪除轉換器呼叫數上限
DescribeDestinations 調節每秒交易的限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒描述目的地呼叫數量上限

名稱	預設	可調整	Description
DescribeExportTasks 每秒交易的調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 describe-export-tasks 呼叫數量上限
DescribeImportTaskBatches 每秒交易的調節限制	每個受支援的區域：10	否	每個帳戶/每個區域的每秒 DescribeImportTaskBatches 呼叫數上限
DescribeImportTaskBatches 暴增中每秒交易的調節限制	每個受支援的區域：10	否	爆量中每個帳戶/每個區域的每秒 DescribeImportTaskBatches 呼叫數量上限
DescribeImportTasks 每秒交易的調節限制	每個受支援的區域：10	否	每個帳戶/每個區域的每秒 DescribeImportTasks 呼叫數量上限
DescribeImportTasks 暴增中每秒交易的調節限制	每個受支援的區域：10	否	高載中每個帳戶/每個區域的每秒 DescribeImportTasks 呼叫數量上限
DescribeLogGroups 每秒交易的調節限制	每個支援的區域：每秒 10	是	每個帳戶/每個區域的每秒 describe-log-groups 呼叫數量上限
每秒交易的 DescribeLogStreams 限流限制	每個受支援的區域：每秒 25 個	是	每個帳戶/每個區域的每秒 describe-log-streams 呼叫數量上限
DescribeMetricFilters 每秒交易的調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域每秒的 describe-metric-filters 呼叫數量上限

名稱	預設	可調整	Description
DescribeSubscriptionFilters 每秒交易的調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 describe-subscription-filters 呼叫數上限
DisassociateSourceFromS3TableIntegration調節限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 DisassociateSourceFromS3TableIntegration 呼叫數上限
DisassociateSourceFromS3TableIntegration調節限制	每個受支援的區域：10	否	高載中每個帳戶/每個區域的每秒 DisassociateSourceFromS3TableIntegration 呼叫數上限
事件大小	每個支援的區域：1，024 KB	否	日誌事件大小上限，以 KB 為單位

名稱	預設	可調整	Description
FilterLogEvents 每秒交易的調節限制	us-east-1：每秒 25 個 ap-northeast-3：每秒 5 個 ap-southeast-3：每秒 5 個 ca-west-1：每秒 5 個 eu-central-1：每秒 5 個 il-central-1：每秒 5 個 每個其他支援的區域：每秒 10 個	否	每個帳戶/每個區域的每秒 filter-log-events 呼叫數量上限
每秒交易的 GetDataProtectionPolicy 調節限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 get-data-protection-policy 呼叫數量上限

名稱	預設	可調整	Description
每秒交易中的 GetLogEvents 調節限制	us-west-2：每秒 10 個 ap-northeast-3：每秒 10 個 ap-southeast-3：每秒 10 個 ca-west-1：每秒 10 個 eu-central-1：每秒 10 個 eu-west-1：每秒 10 個 eu-west-3：每秒 30 個 il-central-1：每秒 10 個 每個其他支援的區域：每秒 25 個	否	每個帳戶/每個區域的每秒 get-log-events 呼叫數量上限
GetQueryResults 每秒交易的調節限制	每個受支援的區域：10	否	每個帳戶/每個區域的每秒 get-query-results 呼叫數量上限
每秒交易中的 GetTransformer 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域每秒的 get-transformer 呼叫數量上限

名稱	預設	可調整	Description
ListSourcesForS3TableIntegration調節限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 ListSourcesForS3TableIntegration 呼叫數目上限
ListSourcesForS3TableIntegration調節限制	每個受支援的區域：10	否	爆量中每個帳戶/每個區域的 ListSourcesForS3TableIntegration 每秒呼叫數上限
每秒交易的 ListTagsForResource 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域每秒的 list-tags-for-resource 呼叫數量上限
每秒交易的 ListTagsLogGroup 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 list-tags-log-group 呼叫數上限
Live Tail 並行工作階段限制	每個受支援的區域：15	是	每個帳戶的並行作用中 Live Tail 工作階段數目上限
日誌群組	每個受支援的區域：1,000,000	是	帳戶可擁有的日誌群組數量上限
每個 Live Tail 工作階段掃描的日誌群組	每個受支援的區域：10	否	每個 Live Tail 工作階段可掃描的日誌群組數量上限
每個日誌群組的指標篩選條件	每個受支援的區域：100	否	每個日誌群組的指標篩選條件數量
每秒交易的 PutBearerTokenAuthentication 限流限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 put-bearer-token-authentication 呼叫數上限

名稱	預設	可調整	Description
PutBearerTokenAuthentication 暴增中每秒交易的限流限制	每個受支援的區域：10	否	爆量中每個帳戶/每個區域的每秒 put-bearer-token-authentication 呼叫數上限
每秒交易的 PutDataProtectionPolicy 限流限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 put-data-protection-policy 呼叫數量上限
每秒交易的 PutDestination 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-destination 呼叫數量上限
每秒交易的 PutDestinationPolicy 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-destination-policy 呼叫數量上限
PutLogEvents 每秒交易的調節限制	每個支援的區域：每秒 5，000 個	是	每個帳戶/每個區域的每秒 put-log-events 呼叫數量上限
每秒交易的 PutMetricFilter 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-metric-filter 呼叫數上限
每秒交易的 PutRetentionPolicy 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-retention-policy 呼叫數量上限
PutSubscriptionFilter限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-subscription-filter 呼叫數上限
每秒交易的 PutTransformer 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 put-transformer 呼叫數量上限

名稱	預設	可調整	Description
資源政策	每個受支援的區域：10	否	每個帳戶/每個區域的資源政策數量上限
每秒交易的 StartLiveTail 限流限制	每個受支援的區域：5	否	每個帳戶/每個區域的每秒 start-live-tail 呼叫數量上限。此限制獨立適用於主控台和 API
每秒交易的 StartQuery 限流限制	每個受支援的區域：10	否	每個帳戶/每個區域的每秒開始查詢呼叫數量上限
每個日誌群組的訂閱篩選條件	每個受支援的區域：2	否	每個日誌群組的訂閱篩選條件數量
TagLogGroup 調節限制，以每秒交易為單位	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 tag-log-group呼叫數量上限
每秒交易的 TagResource 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒標籤資源呼叫數上限
每秒交易的 TestMetricFilter 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 test-metric-filter呼叫數上限
每秒交易的 TestTransformer 限流限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒測試轉換器呼叫數上限
每秒交易的 UntagLogGroup 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒 untag-log-group 呼叫數量上限
每秒交易中的 UntagResource 調節限制	每個支援的區域：每秒 5 個	否	每個帳戶/每個區域的每秒取消標記資源呼叫數量上限

名稱	預設	可調整	Description
記錄異常偵測器	每個受支援的區域：500	是	作用中日誌異常偵測器的數量上限

管理您的 CloudWatch Logs 服務配額

CloudWatch Logs 已與 Service Quotas 整合，此服務 AWS 可讓您從中央位置檢視和管理配額。如需詳細資訊，請參閱《Service Quotas 使用者指南》中的什麼是 Service Quotas？。

Service Quotas 可讓您輕鬆查詢 CloudWatch Logs 服務配額的值。

AWS 管理主控台

使用主控台來檢視 CloudWatch Logs 服務配額

1. 開啟 Service Quotas 主控台，網址為 https://console.aws.amazon.com/servicequotas/。
2. 在導覽窗格中，選擇 AWS services (AWS 服務)。
3. 從 AWS services (AWS 服務) 清單中，搜尋並選取 Amazon CloudWatch Logs。

   在 Service quotas (服務配額) 清單中，您可以看到服務配額名稱、套用的值 (如果有的話)、AWS 預設配額，以及配額值是否可調整。
4. 若要檢視服務配額的其他資訊 (例如說明)，請選擇配額名稱。
5. (選用) 若要請求增加配額，請選取您要增加的配額、選取 Request quota increase (請求增加配額)、輸入或選取必要資訊，然後選取 Request (請求)。

若要使用主控台來進一步處理服務配額，請參閱《Service Quotas 使用者指南》。若要請求提高配額，請參閱《Service Quotas 使用者指南》中的請求提高配額。

AWS CLI

使用 檢視 CloudWatch Logs 服務配額 AWS CLI

執行下列命令，以檢視預設 CloudWatch Logs 配額。

```
aws service-quotas list-aws-default-service-quotas \
```

```
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code logs \
    --output table
```

若要使用 進一步使用服務配額 AWS CLI，請參閱 Service Quotas AWS CLI 命令參考。若要請求提高配額，請參閱《AWS CLI 命令參考》中的 `request-service-quota-increase` 命令。

文件歷史紀錄

下表說明從 2018 年 6 月開始，每一版《CloudWatch Logs 使用者指南》的重要變更。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

變更	描述	日期
使用 CloudWatch Logs Insights 參數儲存的查詢	您現在可以在 CloudWatch Logs Insights 中使用具名參數建立可重複使用的查詢範本。定義單一範本並在執行時間傳遞不同的值，而不是維護多個幾乎相同的已儲存查詢。如需詳細資訊，請參閱搭配參數使用已儲存的查詢。	2026 年 3 月 26 日
新的 CloudWatch Logs 受管政策 CloudWatchLogsAPIKeyAccess	CloudWatch Logs 已發佈新的受管政策`CloudWatchLogsAPIKeyAccess`，啟用 CloudWatch Logs API 金鑰身分驗證和加密日誌擷取。如需詳細資訊，請參閱 CloudWatch Logs 受 AWS 管政策的更新。	2026 年 2 月 17 日
已更新 CloudWatchLogsReadOnlyAccess 政策	可觀測性管理動作的許可已新增至 CloudWatchLogsReadOnlyAccess，以允許唯讀存取遙測管道和 S3 資料表整合。	2025 年 12 月 2 日
已更新 CloudWatchLogsFullAccess 政策	可觀測性管理動作的許可已新增至 CloudWatchLogsFullAccess，以允許唯讀存取遙測管道和 S3 資料表整合。	2025 年 12 月 2 日
CloudWatch Logs 中的新日誌管理功能	CloudWatch 將日誌管理合併為具有內建控管功能的單一服	2025 年 12 月 2 日

	務，無需在不同工具和資料存放區中存放和維護相同資料的多個副本。如需詳細資訊，請參閱日誌管理。	
CloudWatch Logs 中的新資料擷取和標準化功能	CloudWatch 透過整合 AWS Organizations 和預先建置的第三方來源連接器，自動收集跨帳戶和 AWS 區域的 AWS 付費日誌。如需詳細資訊，請參閱在擷取期間轉換日誌。	2025 年 12 月 2 日
CloudWatch Logs Insights 中根據資料來源功能進行的新分析和查詢	CloudWatch Insights 介紹查詢的面向。面向有助於分析日誌，因為它們可讓您以互動方式篩選和分組資料，而無需執行查詢。面向是日誌中的欄位（例如 ServiceName 或 StatusCode），可跨日誌群組進行篩選、彙總和分析。如需詳細資訊，請參閱使用面向來分組和探索日誌。	2025 年 12 月 2 日
新的 S3 資料表整合	S3 Tables 與 CloudWatch 整合可讓您使用分析引擎存取擷取至 CloudWatch 的日誌資料，例如 Amazon Athena、Amazon Redshift，以及支援連線至 Apache Iceberg 相容存放區的第三方工具。如需詳細資訊，請參閱使用 S3 Tables 整合存取日誌。	2025 年 12 月 2 日

CloudWatch Logs 新增對跨帳戶跨區域日誌集中化的支援	CloudWatch Logs 新增對跨帳戶跨區域日誌集中化的支援，讓 CloudWatch Logs 能夠將 AWS Organizations 成員帳戶中的日誌資料收集到中央帳戶以進行分析。如需詳細資訊，請參閱跨帳戶跨區域日誌集中。	2025 年 9 月 17 日
CloudWatch Logs 異常偵測已更新範例政策	CloudWatch Logs 更新了 KMS 範例政策，透過新增 和 的條件來縮小其範圍`aws:SourceAccount` 並改善 的安全性`aws:SourceArn` 。如需詳細資訊，請參閱使用 AWS KMS 加密異常偵測器及其結果	2025 年 7 月 14 日
CloudWatch Logs Insights 新增對 Amazon VPC Route Server 日誌的支援	CloudWatch Logs Insights 新增對 Amazon VPC Route Server 日誌的支援。如需詳細資訊，請參閱從 AWS 服務啟用記錄。Amazon VPC Route Server 的新日誌來源`EVENT_LOGS` 記錄在 PutDeliverySource 中	2025 年 6 月 12 日
CloudWatch Logs Insights 新增對 AWS PCS 日誌的支援	CloudWatch Logs Insights 新增對 AWS PCS 日誌的支援。如需詳細資訊，請參閱從 AWS 服務啟用記錄。 AWS PCS`PCS_SCHEDULER_LOGS` 和 的新日誌來源`PCS_JOBCOMP_LOGS` 記錄在 PutDeliverySource 中	2025 年 6 月 12 日

CloudWatch Logs Insights 新增對 AWS Entity Resolution 日誌的支援	CloudWatch Logs Insights 新增對 AWS Entity Resolution 日誌的支援。如需詳細資訊，請參閱從 AWS 服務啟用記錄。PutDeliverySource 中 AWS Entity Resolution WORKFLOW_LOGS 記錄了 的新日誌來源	2025 年 5 月 22 日
CloudWatch Logs 受管政策更新以支援自然語言摘要	的許可cloudwatch:GenerateQueryResultsSummary 已新增至 CloudWatchLogsFullAccess 和 CloudWatchLogsReadOnlyAccess，允許產生查詢結果的自然語言摘要。若要查看政策的內容，請參閱《AWS 受管政策參考指南》中的 CloudWatchLogsFullAccess 和 CloudWatchLogsReadOnlyAccess。	2025 年 5 月 20 日
CloudWatch Logs Insights 新增支援從 CloudWatch Logs Insights 查詢結果產生自然語言摘要	新增對 CloudWatch Logs Insights 中自然語言摘要的支援。此功能會產生查詢結果的人類可讀摘要，目前在美國東部（維吉尼亞北部）提供。如需詳細資訊，請參閱從 CloudWatch Logs Insights 查詢結果產生自然語言摘要。	2025 年 5 月 20 日

CloudWatchOpenSearchDashboardAccess 和 CloudWatchOpenSearchDashboardsFullAccess 是新的 IAM 政策	CloudWatch Logs 新增了兩個新的 IAM 政策：CloudWatchOpenSearchDashboardsFullAccess 和 CloudWatchOpenSearchDashboardAccess。CloudWatchOpenSearchDashboardsFullAccess 授予建立和管理與 OpenSearch Service 整合的許可。CloudWatchOpenSearchDashboardAccess 授予檢視在這些整合中建立之已結束日誌儀表板的存取權。如需詳細資訊，請參閱由 Amazon OpenSearch Service 提供支援的已終止日誌儀表板。	2024 年 12 月 1 日
CloudWatchLogsFullAccess 政策已更新	CloudWatch Logs 將 Amazon OpenSearch Service 和 IAM 的許可新增至 CloudWatchLogsFullAccess 政策，以針對某些功能啟用 CloudWatch Logs 與 OpenSearch Service 的整合。	2024 年 12 月 1 日
CloudWatch Logs Insights 會將新的結構類型新增至查詢語法	CloudWatch Logs Insights 新增 `unnest`命令和兩個 JSON 函數，可讓您操作 JSON 字串做為映射和清單。如需詳細資訊，請參閱結構類型。	2024 年 11 月 21 日

CloudWatch Logs 在日誌擷取期間支援日誌轉換	您可以建立日誌轉換器，在擷取時修改日誌事件，協助您將日誌以不同格式從不同來源標準化為一致且內容豐富的格式。如需詳細資訊，請參閱在擷取期間轉換日誌。	2024 年 11 月 20 日
CloudWatch Logs Insights 新增欄位索引	CloudWatch Logs Insights 已新增對日誌欄位索引的支援。當您接著在 CloudWatch Logs Insights 查詢中使用欄位索引時，查詢會嘗試略過處理已知不包含索引欄位的日誌事件。如需詳細資訊，請參閱建立欄位索引以改善查詢效能並減少掃描磁碟區。	2024 年 11 月 20 日
CloudWatch Logs Insights 支援自然語言查詢產生	CloudWatch Logs Insights 支援自然語言來產生和更新查詢。如需詳細資訊，請參閱使用自然語言來產生和更新 CloudWatch Logs Insights 查詢。	2024 年 6 月 20 日
CloudWatchLogsReadOnlyAccess 政策已更新	CloudWatch Logs 已將 `cloudwatch:GenerateQuery` 許可新增至 CloudWatchLogsReadOnlyAccess，因此使用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。	2023 年 11 月 26 日

CloudWatchLogsFullAccess 政策已更新	CloudWatch Logs 已將 `cloudwatch:GenerateQuery` 許可新增至 CloudWatchLogsFullAccess，因此使用此政策的使用者可以從自然語言提示產生 CloudWatch Logs Insights 查詢字串。	2023 年 11 月 26 日
CloudWatch Logs 新增日誌模式分析	CloudWatch Logs 現在會在您每次執行 CloudWatch Logs Insights 查詢時掃描日誌事件中的模式。如需詳細資訊，請參閱模式分析。	2023 年 11 月 26 日
CloudWatch Logs 新增日誌異常偵測	您可以為日誌群組建立日誌異常偵測器。異常偵測器會掃描擷取至日誌群組的日誌事件，並在日誌資料中尋找異常。如需詳細資訊，請參閱日誌異常偵測。	2023 年 11 月 26 日
CloudWatch Logs 新增比較功能	您現在可以使用 CloudWatch Logs Insights 來比較日誌事件隨時間的變化。如需詳細資訊，請參閱比較（差異）與先前的時間範圍。	2023 年 11 月 26 日
CloudWatch Logs 新增新的日誌類別	CloudWatch Logs 支援兩種類別的日誌群組，因此您可以針對不常存取的日誌使用符合成本效益的選項，而且對於需要即時監控或其他功能的日誌，您也可以使用功能完整的選項。如需詳細資訊，請參閱日誌類別。	2023 年 11 月 26 日

CloudWatch Logs Insights 支援自然語言查詢產生	CloudWatch Logs Insights 支援自然語言來產生和更新查詢。如需詳細資訊，請參閱使用自然語言來產生和更新 CloudWatch Logs Insights 查詢。	2023 年 11 月 26 日
CloudWatch Logs 新增支援 Live Tail 的規則表達式篩選條件模式語法	現在，您可以在 Live Tail 篩選條件模式中使用靈活的規則表達式，進一步自訂搜尋和比對操作，以滿足您的需求。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的篩選條件模式語法。	2023 年 11 月 13 日
CloudWatch Logs 新增支援指標篩選條件、訂閱篩選條件和篩選條件日誌事件的規則表達式篩選模語法	現在，您可以在篩選條件模式中使用靈活的規則運算式，進一步自訂搜尋和比對操作，以滿足您的需求。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的篩選條件模式語法。	2023 年 9 月 5 日
CloudWatch Logs Insights 新增 pattern 命令	您現在可以在 CloudWatch Logs Insights 查詢中使用 pattern，將日誌資料自動叢集化，形成模式。模式是指日誌欄位之間反覆出現的共同文字結構。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 pattern。	2023 年 7 月 17 日

CloudWatch Logs Insights 新增了一個 dedup 命令	現在可以在 CloudWatch Logs Insights 查詢中使用 dedup，根據您指定欄位中的特定值移除重複的結果。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南中的 dedup。	2023 年 6 月 20 日
帳戶層級資料保護政策	您現在可以在帳戶層級設定資料保護政策。這些帳戶層級政策可以稽核和遮罩帳戶中所有日誌群組中日誌事件中的敏感資訊。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南中的使用遮罩功能協助保護敏感日誌資料。	2023 年 6 月 8 日
新增了 Live Tail 功能	CloudWatch Logs 新增了 Live Tail 功能，因此您可以在擷取日誌時進行掃描，以協助進行疑難排解。您可以選擇性地根據指定的詞彙來篩選顯示的日誌事件串流，也可以反白顯示包含指定詞彙的日誌事件。如需詳細資訊，請參閱使用 Live Tail 以近乎實時的方式檢視日誌。	2023 年 6 月 6 日

CloudWatchLogsReadOnlyAccess 政策已更新	CloudWatch Logs 已將許可新增至 CloudWatchLogsReadOnlyAccess。已新增 `logs:StartLiveTail` 和 `logs:StopLiveTail` 許可，以便具有此政策的使用者可以使用主控台啟動和停止 CloudWatch Logs Live Tail 工作階段。如需詳細資訊，請參閱使用 Live Tail 以近乎即時的方式檢視日誌。	2023 年 6 月 6 日
CloudWatch Logs Insights 已發行	您可以使用 CloudWatch Logs Insights 以互動方式搜尋和分析日誌資料。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的使用 CloudWatch Logs Insights 分析日誌資料。	2018 年 11 月 27 日
支援 Amazon VPC 端點	您現在可以在 VPC 和 CloudWatch Logs 之間建立私有連線。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 CloudWatch Logs 搭配界面 VPC 端點一起使用。	2018 年 6 月 28 日

下表說明《Amazon CloudWatch Logs 使用者指南》的重要變更。

變更	Description	發行日期
介面 VPC 端點	在某些區域，您可以使用界面 VPC 端點，避免 Amazon VPC 和 CloudWatch Logs 之間的流量離開 Amazon 網路。如需詳細資訊，請參閱 搭配介面 VPC 端點使用 CloudWatch Logs。	2018 年 3 月 7 日

變更	Description	發行日期
Route 53 DNS 查詢日誌	您可以使用 CloudWatch Logs 來存放 Route 53 所接收 DNS 查詢的相關日誌。如需詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的什麼是 Amazon CloudWatch Logs？或記錄 DNS 查詢。	2017 年 9 月 7 日
標記日誌群組	您可以使用標籤來分類您的日誌群組。如需詳細資訊，請參閱在 Amazon CloudWatch Logs 中標記日誌群組。	2016 年 12 月 13 日
主控台改進	您可以從指標圖表導覽到關聯的日誌群組。如需詳細資訊，請參閱從指標轉換到日誌。	2016 年 11 月 7 日
主控台可用性改善	提升經驗，讓您更輕鬆地搜尋、篩選及進行故障診斷。例如，現在您可以篩選特定日期和時間範圍的日誌資料。如需詳細資訊，請參閱檢視傳送到 CloudWatch Logs 的日誌資料。	2016 年 8 月 29 日
新增對 Amazon CloudWatch Logs 和新 CloudWatch Logs 指標的 AWS CloudTrail 支援	新增對 CloudWatch Logs 的 AWS CloudTrail 支援。如需詳細資訊，請參閱在 中記錄 CloudWatch Logs API 和主控台操作 AWS CloudTrail。	2016 年 3 月 10 日
新增支援 CloudWatch Logs 匯出至 Amazon S3	新增支援將 CloudWatch Logs 資料匯出至 Amazon S3。如需詳細資訊，請參閱將日誌資料匯出至 Amazon S3。	2015 年 12 月 7 日
新增對 Amazon CloudWatch Logs 中 AWS CloudTrail 記錄事件的支援	您可以在 CloudWatch 中建立警示，並接收 CloudTrail 所擷取特定 API 活動的通知，然後使用此通知進行故障診斷。	2014 年 11 月 10 日

變更	Description	發行日期
新增對 Amazon CloudWatch Logs 的支援	您可以使用 Amazon CloudWatch Logs，以監控、存放和存取來自 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或其他來源的系統、應用程式和自訂日誌檔案。然後，您可以使用 Amazon CloudWatch 主控台、 中的 CloudWatch Logs 命令 AWS CLI或 CloudWatch Logs SDK，從 CloudWatch Logs 擷取相關聯的日誌資料。如需詳細資訊，請參閱什麼是 Amazon CloudWatch Logs？。	2014 年 7 月 10 日

AWS 詞彙表

如需最新的 AWS 術語，請參閱 AWS 詞彙表 參考中的AWS 詞彙表。

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。